DE 9-1995-0084

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 197 18 115 A 1**

(51) Int. Cl.[6]:
G 06 K 19/07
G 07 F 7/08

(21) Aktenzeichen: 197 18 115.5
(22) Anmeldetag: 29. 4. 97
(43) Offenlegungstag: 25. 6. 98

DE 197 18 115 A1

(66) Innere Priorität:
196 54 187. 5 23. 12. 96

(71) Anmelder:
CCS Chipcard & Communication Systems GmbH, 81539 München, DE

(74) Vertreter:
Betten & Resch, 80469 München

(72) Erfinder:
Burger, Adelheid, 64625 Bensheim, DE; Engelhardt, Holger, 13156 Berlin, DE; Hinz, Michael, 34277 Fuldabrück, DE; Kissinger, Stefan, Dr., 10823 Berlin, DE; Gollner, Michael, Dr., 81549 München, DE; Kuchelmeister, Anton J., Dr., 80995 München, DE; Schwier, Andreas, 32429 Minden, DE; Radnoti, Michael, 85417 Marzling, DE

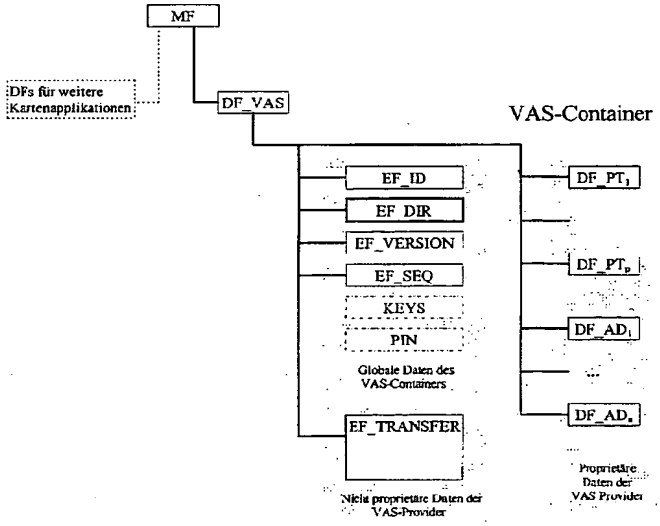
(56) Für die Beurteilung der Patentfähigkeit in Betracht zu ziehende Druckschriften:

DE	41 19 924 C3
DE	39 06 349 C2
DE	38 12 147 C2
DE	195 22 050 A1
DE	195 22 029 A1
DE	43 33 388 A1
DE	41 15 152 A1
DE	38 44 032 A1
DE	38 07 997 A1
US	55 42 081
US	52 52 812
EP	06 46 898 A2
EP	05 83 006 A2

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Chipkarte und Verfahren zur Verwendung der Chipkarte

(57) Chipkarte, die zur Durchführung von Transaktionen dient, bei denen geldwerte Einheiten oder andere nichtmonetäre Ansprüche repräsentierende Wertdaten zwischen dem Karteninhaber und mindestens einem Transaktionspartner (Service-Provider) übertragen oder dem Service-Provider zur Verifizierung der Ansprüche vorgewiesen werden, wobei die Chipkarte einen Speicher umfaßt, in dem zur Durchführung der Transaktionen erforderliche Daten abgespeichert werden, und die Chipkarte ferner folgendes aufweist: eine Einrichtung zum Laden von einer oder mehreren Kartenanwendungen (VAS-Applikationen) auf die Karte, die jeweils die Durchführung von Transaktionen zwischen dem Karteninhaber und einem oder mehreren Service-Providern ermöglichen.



DE 197 18 115 A1

Beschreibung

Die Erfindung betrifft eine Chipkarte, ein Terminal zur Verwendung mit einer Chipkarte, ein Verfahren zur Verwendung der Chipkarte sowie ein Chipkartensystem.

Es sind bereits Mikroprozessor-Chipkarten mit Zahlungsfunktion, z. B. elektronische Geldbörsen, ec-Cash, Kreditfunktion usw., im Einsatz und es sind je nach Ausprägung durch Organisationen, wie ZKA (Zentraler Kreditausschuß), VISA oder EMV (Europay Mastercard VISA), Vorgaben festgelegt, so daß sie als Zahlungsmittel ("Geldersatz") verwendet werden können. Als beispielhafte Beschreibungen seien hier genannt:

- ZKA, Zentraler Kreditausschuß, "Schnittstellenspezifikation für die ec-Karte mit Chip", 27.10.1995;
- Europay International, "Integrated Circuit Card, Specification for Payment Systems, EMV'96", V3.0, 30-Jun-1996;
- ISO/IEC 7816-4 "Information technology – Identification cards – Integrated circuit(s) cards with contacts, Part 4: Interindustry commands for interchange", 01-09-1995;
- prEN 1546-1, "Identification card systems – Inter-sector electronic purse, Part 1: Definitions, concepts and structures", 15.03.1995;
- prEN 1546-2, "Identification card systems – Inter-sector electronic purse, Part 2: Security architecture", 03.07.1995;
- prEN 1546-3, "Identification card systems – Inter-sector electronic purse, Part 3: Data elements and interchanges", 09.12.1994.

Ein aktueller Überblick über Chipkarten ist zu finden in:

- Stefan Schütt, Bert Kohlgraf: "Chipkarten, Technische Merkmale, Normung, Einsatzgebiete", R. Oldenbourg Verlag, München/Wien, 1996, ISBN 3-486-23738-1.

Herkömmliche Chipkarten sind regelmäßig nur für einen bestimmten Anwendungszweck, beispielsweise als elektronische Geldbörse oder als elektronischer Ausweis, nutzbar. Die auf diesen Chipkarten aufgebrachten Anwendungen sind jedoch regelmäßig statisch, d. h. sie werden bei der Herstellung der Chipkarte aufgebracht und bleiben über den Lebenszyklus der Karte hinweg unverändert bestehen.

Herkömmliche Chipkarten sind also sowohl hinsichtlich ihrer Variabilität als auch hinsichtlich ihrer Funktionalität beschränkt. Insbesondere sind herkömmliche Chipkarten nach dem Herstellungsprozeß hinsichtlich ihrer Funktionalität festgelegt und nicht mehr veränderbar.

Es ist daher eine Aufgabe der vorliegenden Erfindung, eine Chipkarte zu schaffen, deren Funktionalität variabel ist.

Eine weitere Aufgabe der Erfindung besteht darin, eine Chipkarte zu schaffen, bei der die Zahl und Art der mit der Chipkarte durchführbaren Applikationen bzw. Anwendungen und Transaktionen auch nach dem Herstellungsprozeß noch variabel ist. Es soll möglich sein, auf diese Chipkarte zusätzliche Applikationen "zu laden", es sollen Applikationen von der Chipkarte gelöscht werden können und die einzelnen Applikationen sollen daten- und sicherheitstechnisch unabhängig voneinander definiert sein und unabhängig voneinander ablaufen.

Die Chipkarte soll beispielsweise die daten- und sicherheitstechnischen Voraussetzungen nach ISO 7816 erfüllen; die einzelnen Applikationen der Karte sollen jedoch insbesondere von der Kartenplattform selbst unabhängig sein.

Es ist eine weitere Aufgabe der Erfindung, eine Chipkarte zu schaffen, bei der der Benutzer die Anzahl und Art der in seiner Karte verfügbaren Applikationen selbst bestimmen bzw. zusammenstellen und ändern kann.

Es ist ferner eine Aufgabe der vorliegenden Erfindung, eine Chipkarte zu schaffen, mit der sowohl Intraservices (d. h. geschlossene Applikationen in dem Sinne, daß keine Abrechnung und Leistungsübertragung mit externen Partnern zu erfolgen hat) als auch Interservices (d. h. Applikationen mit zusätzlichen Außenbeziehungen zu externen Partnern) ermöglicht und durchgeführt werden können.

Gemäß einem Aspekt der Erfindung ist eine Einrichtung vorgesehen, mit der eine oder mehrere Kartenanwendungen auf die Karte geladen werden können, mit denen jeweils die Durchführung von Transaktionen zwischen dem Karteninhaber und einem oder mehreren Service-Providern ermöglicht wird.

Durch das Laden wird die Chipkarte so konfiguriert, daß sie eine neue Funktionalität erhält, d. h. eine Applikation ausführen kann, die ihr bisher nicht möglich war. Durch die geladenen Daten werden Applikationen definiert und in Verbindung mit den Grundfunktionalitäten der Karte wie etwa dem Betriebssystem realisiert, die vorher nicht auf der Karte vorhanden waren. Damit wird die Funktionalität der Karte durch das Laden einer Applikation um eben diese Applikation erweitert.

Gemäß einem Ausführungsbeispiel der Erfindung ist auf der erfindungsgemäßen Chipkarte eine Datenstruktur (DF_VAS) vorgesehen, die selbst wiederum in eine Teilstruktur und einen Definitionsdatensatz unterteilt ist, wobei die Datenstruktur mittels einer sie identifizierenden Kennung eindeutig identifiziert ist und somit von der Kartenplattform an sich unabhängig ist. In die Teilstruktur können nun sogenannte Applikationen geladen werden, d. h. Funktionalitäten oder Anwendungen der Chipkarte. Damit wird es durch Laden einer bestimmten Applikation der Chipkarte möglich, Transaktionen zwischen dem Karteninhaber und einem für diese Applikation spezifischen Service-Provider durchzuführen. Ein Definitionsdatensatz innerhalb der Datenstruktur enthält Informationen über die Art und/oder die Struktur der in der Teilstruktur geladenen Applikationen. Zumindest der Definitionsdatensatz, vorzugsweise jedoch die gesamte Datenstruktur sind mittels mindestens eines Systemschlüssels gegen Modifikationen abgesichert und nur unter Verwendung dieses Schlüssels modifizierbar. Anstelle eines Systemschlüssels sind auch andere Sicherungsmechanismen oder Sicherungseinrichtungen vorstellbar, mittels derer die Absicherung gegen Modifikationen erfolgen kann, etwa durch eine persönliche Kennziffer (PIN) oder sonstige zur Absicherung dienende Einrichtungen.

Mit der obigen Struktur ist es möglich, in die Teilstruktur verschiedene Applikationen zu laden und diese auch wieder

aus der Teilstruktur zu löschen, so daß die Karte hinsichtlich ihrer Funktionalitäten bzw. der mit ihr durchführbaren Anwendungen variabel ist. Laden und Löschen von Applikationen geschieht durch Schreiben von applikationsspezifischen Daten und Schlüsseln in die vorhandene Teilstruktur. Durch Vorsehen des Systemschlüssels und der Datenstrukturkennung wird die die Multifunktionalität ermöglichende Datenstruktur unabhängig von der Kartenplattform an sich und auch hinsichtlich ihrer Sicherheitsarchitektur selbsttragend und unabhängig von der Kartenplattform.

Abhängig von den in der Teilstruktur geladenen Applikationen können dann mittels der Karte Transaktionen zwischen dem Karteninhaber und Service-Providern, die von den geladenen Applikationen abhängig sind, durchgeführt werden.

Vorzugsweise weist die Chipkarte ferner einen Transfer-Speicherbereich auf, in den die bei der Durchführung von Transaktionen auszutauschenden Daten geschrieben bzw. aus dem sie gelesen werden. Indem für das Entnehmen von Daten aus dem Transfer-Speicherbereich terminalspezifische Schlüssel vorgesehen werden, sind die einzelnen Zugriffe sicherheitstechnisch voneinander unabhängig.

Die einzelnen in der Karte vorhandenen Teilstrukturen bzw. Applikationen sind vorzugsweise voneinander unabhängig und jeweils einem bestimmten Service-Provider zugeordnet. Sie stellen sozusagen die für diesen Service-Provider proprietären oder spezifischen Daten zum Durchführen einer bestimmten Applikation dar. Sie enthalten deshalb je nach Applikationstyp und je nach Service-Provider unterschiedliche Informationen, beispielsweise Daten, die einen bestimmten Wert repräsentieren (Bonuspunkte, Kontostand, etc.), Informationsdaten über die Applikation, Informationsdaten über den Service-Provider, etc. Vorzugsweise enthalten sie jedoch insbesondere auch für die Applikation spezifische Schlüssel, mittels derer ein Zugriff auf die Daten der Teilstruktur auf sicherheitstechnisch von anderen Teilstrukturen unabhängige Art und Weise ermöglicht wird. Das Laden oder Generieren der Teilstruktur bzw. Applikation selbst ist dagegen mit zumindest einem übergeordneten Systemschlüssel abgesichert.

Vorzugsweise findet vor dem Durchführen einer Transaktion eine gegenseitige Authentifizierung zwischen Chipkarte und einem Terminal statt, wobei dabei ein applikations- bzw. teilstrukturspezifischer Authentifizierungsschlüssel vorgesehen ist.

Zum Durchführen von Transaktionen werden dann Daten in eine für die jeweilige VAS-Applikation reservierte Teilstruktur geschrieben oder daraus gelesen oder über den Transfer-Speicherbereich abgewickelt. Im ersten Fall werden applikationsspezifische Schlüssel verwendet. Im letzteren Fall wird ein applikationsspezifischer und vorzugsweise auch ein terminalspezifischer Schlüssel, der beispielsweise mittels eines Schlüssel-Erzeugungsschlüssels, der auf der Karte vorgesehen ist und aus applikationsspezifischen Daten generiert wird, verwendet. Die so in den Transferspeicher geschriebenen Daten können dann vom Service-Provider über das Terminal entnommen werden, was vorzugsweise durch Kennzeichnen der im Transferspeicher gespeicherten Daten als entwertet geschieht. Handelt es sich dabei um einen Service-Provider, der nicht für die schreibende Applikation spezifisch ist, so wird dadurch ein sogenannter Interservice ausgeführt, d. h. es werden zum Nutzen bzw. auf Kosten des Karteninhabers geldwerte Daten zwischen unterschiedlichen Service-Providern ausgetauscht.

Als zusätzliche Sicherung ist ferner eine PIN oder ein Paßwort zur Verifikation der Berechtigung eines Transaktionsvorgangs vorgesehen.

Durch die variable Struktur der Karte können zu verschiedenen Zeitpunkten verschiedene Applikationen in unterschiedlicher Anzahl auf der Karte untergebracht sein.

Die Echtheit von aus dem Transfer-Speicher entnommenen Daten wird vorzugsweise unter Verwendung eines Signierschlüssels bzw. unter Verwendung einer digitalen Unterschrift oder mindestens eines Schlüssels gesichert. Besonders vorteilhaft ist es dabei, wenn die entnommenen Daten weiter im Transfer-Speicher verbleiben, jedoch lediglich durch die Karte als entnommen gekennzeichnet werden. Dadurch können auch nach der Transaktion noch Informationen über die durchgeführte Transaktion erhalten werden. Damit und mit der Absicherung der Entnahme ist auch Einmaligkeit nachweisbar.

Besonders vorteilhaft ist es ferner, wenn die in den Transfer-Speicher geschriebenen Daten mit einem Verfallsdatum gekennzeichnet sind, nach dem sie ihren Wert verlieren. Dadurch werden Applikationen realisierbar, die beispielsweise die Bereitstellung eines für einen bestimmten Zeitraum gültigen Tickets ermöglichen.

Mittels eines vorzugsweise vorgesehenen Transaktionszählers können die Transaktionsdaten bzw. die Wertdaten hinsichtlich ihrer zugehörigen Transaktion eindeutig bestimmt und identifiziert werden.

Die erfindungsgemäße Chipkarte besteht in einer vorteilhaften Ausführungsform also aus einem hierarchischen Speicherkonzept, das auf seinen unterschiedlichen Ebenen mittels verschiedener Schlüssel gegen Modifikationen abgesichert ist, das auf der Ebene der Applikationen hinsichtlich der in den Speicher geladenen Applikationen variabel ist, wobei jede einzelne Applikation durch eigene spezifische Schlüssel gegenüber anderen abgesichert und von diesen unabhängig ist, und wobei die Gesamtstruktur mittels mindestens eines Systemschlüssels und mittels einer die Struktur identifizierenden Kennung gesichert und von der Kartenplattform selbst unabhängig ist. Das Konzept des Transferspeichers ermöglicht den Austausch von Daten sowohl zwischen Karteninhaber und Service-Provider als auch zwischen unterschiedlichen Service-Providern selbst. Auch das Lesen und Schreiben in den bzw. aus dem Transferspeicher ist durch Schlüssel abgesichert, wobei diese ebenfalls karten- und appplikationsspezifisch, zusätzlich jedoch auch noch terminalspezifisch sind. Eine Authentifizierung, die vor jeder Transaktion durchgeführt wird, sowie optional eine PIN oder ein Paßwort erhöhen zusätzlich die Sicherheit der erfindungsgemäßen Chipkarte.

In den Patentansprüchen 21 bis 30 wird ein Terminal zur Verwendung mit der erfindungsgemäßen Chipkarte definiert. Dieses Terminal dient zum Laden oder Löschen von Applikationen, zur Durchführung von Transaktionen, zum Ansehen von Daten sowie zur Durchführung von weiteren in Verbindung mit den jeweiligen Applikationen und Transaktionen stehenden Funktionen. Ein Verfahren zum Durchführen von Transaktionen zwischen Karteninhaber und Service-Provider ist in den Ansprüchen 31 bis 33 definiert, das Laden von Daten auf eine erfindungsgemäße Chipkarte definieren Ansprüche 34 und 35, und Anspruch 36 definiert ein Gesamtsystem aus Chipkarte und Terminal.

Nachfolgend wird die vorliegende Erfindung anhand eines bevorzugten Ausführungsbeispiels näher beschrieben, wobei Bezug auf die beiliegenden Zeichnungen genommen wird. Dabei zeigen

Fig. 1 schematisch die erfindungsgemäße Chipkarte,

Fig. 2 eine schematische Darstellung des Gesamtsystems der Komponenten der Erfindung,

Fig. 3 den Datenfluß im Gesamtsystem,

Fig. 4 die möglichen Applikationsklassen und Operationen durch ein Transaktionsmodell in schematischer Darstellung,

Fig. 5 eine schematische Darstellung der Sicherheitsarchitektur der erfindungsgemäßen Chipkarte,

Fig. 6 die Dateistruktur einer allgemeinen Implementierungsklasse des VAS-Containers,

Fig. 7 verschiedene Implementierungsklassen des VAS-Containers gemäß einem Ausführungsbeispiel der vorliegenden Erfindung,

Fig. 8 die Dateistruktur des VAS-Containers gemäß einem Ausführungsbeispiel der vorliegenden Erfindung,

Fig. 9 schematisch die Dateistruktur der Implementierungsklasse DF_PT,

Fig. 10 schematisch die Dateistruktur der Implementierungsklasse DF_AD.

Bevor die Erfindung näher beschrieben wird, sollen einige, im nachfolgenden verwendeten Begriffe definiert werden:

VAS: Value Added Services

VAS-Karte: Die VAS-Karte ist eine Chipkarte, mit der an den Value Added Services teilgenommen werden kann. Die VAS-Karte enthält neben anderen Anwendungen wie z. B. Zahlungsapplikationen (also elektronische Geldbörse) den VAS-Container.

VAS-Container: Der VAS-Container beinhaltet Datenstrukturen, Zugriffsbedingungen, Schlüssel und (Ergänzungs-) Kommandos zum Verwalten von VAS-Applikationen und der Bereitstellung der Funktionalität der VAS-Applikationen.

VAS-Applikation: VAS-Applikationen enthalten VAS-Daten. Zugriff auf die VAS-Daten wird durch die VAS-Applikation gesteuert. Ein VAS-Provider betreibt im VAS-Container eine oder mehrere VAS-Applikationen. Die Nutzung der VAS-Applikation definiert sich aus dem Aufbringen, Lesen und Verarbeiten von VAS-Daten. Eine VAS-Applikation kann entweder als Intra- oder Interservice ausgeprägt sein.

VAS-Provider: Der VAS-Provider ist für seine VAS-Applikation verantwortlich, die er nach Rahmenbedingungen des System Operators und nach seinen eigenen Vorstellungen entwickelt und danach über den System Operator und die Terminals den Cardholders zur Verwendung bereitstellt. Die VAS-Applikationen sind in den VAS-Container der VAS-Karte zu laden, bevor daran teilgenommen werden kann.

Intraservice: Intraservice ist eine Sorte von VAS-Applikation, die unter exklusiver Regie des jeweiligen VAS-Providers benutzt wird. Intraservice Applikationen sind geschlossene Applikationen in dem Sinne, daß keine Abrechnung oder Leistungsübertragung mit externen Partnern erfolgt. Eine VAS-Applikation kann entweder als Intra- oder Interservice ausgeprägt sein.

Interservices: Interservice Applikationen sind Intraservice Applikationen, die über zusätzliche Außenbeziehungen zu externen Partnern unterhalten. Eine VAS-Applikation kann entweder als Intra- oder Interservice ausgeprägt sein.

SO, System Operator: Der System Operator, oder System Betreiber, bietet den VAS-Providern und den Cardholders das VAS-System zur Nutzung an.

Issuer: Der Issuer, oder Kartenherausgeber, bringt die VAS-Karten mit VAS-Container in den Umlauf.

CH, Cardholder: Der Cardholder, oder Karteninhaber, ist die Person, die die Karte (hier: VAS-Karte) besitzt und einsetzt, um an den Value Added Services teilzunehmen. Diese Person ist nicht zwangsläufig der eigentliche Eigentümer der Karte.

Serviceterminal: Das Serviceterminal wird vom System Operator für VAS-Applikationen aufgestellt. Am Serviceterminal kann der Karteninhaber die VAS-Applikationen auf seiner VAS-Karte verwalten (Laden, Ansehen, Löschen und Übertragen von VAS-Applikationen).

Händlerterminal: Das Händlerterminal besitzt Zahlungsfunktionen und bietet zusätzlich VAS-Funktionalität. An ihm setzt der Karteninhaber seine VAS-Karte ein, um einerseits zu bezahlen und gleichzeitig an den Vorteilen von VAS teilzunehmen.

AID: (Application Identifier) maximal 16 Byte langer Name von Applikationen zur eindeutigen Unterscheidung von Applikationen und der Applikationsselektion von außen ohne Kenntnis der Dateistruktur einer Karte. Die AID besteht aus einer 5 Byte langen Registered Application Provider ID (RID) und optional aus einer maximal 11 Byte langen Proprietary Application Identifier Registration (PIX).

DF: Directory File nach ISO 7816

EF: Elementary File nach ISO 7816

Gültiger VAS-Container: VAS-Container, der sich gegenüber der externen Welt authentisieren kann.

KID: (Key Identifier) Nummer eines Schlüssels innerhalb eines Elementary Files, das Schlüssel enthält

R&R: Rules and Regulations

p: Maximale Zahl von Objekten der Implementierungsklasse DF_PT

a: Maximale Zahl von Objekten der Implementierungsklasse DF_AD

nr_{DIR}: Maximale Gesamtzahl von Objekten: $nr_{DIR} = p + a$. Die Anzahl von Records in EF_DIR ist nr_{DIR}.

$nr_{EF\_TRANSFER}$: Anzahl von Records des EF_TRANSFER.

Fig. 1 zeigt schematisch die Struktur der erfindungsgemäßen Chipkarte. Zusätzlich zu den statisch und nicht veränderbaren, beim Herstellungsprozeß aufgebrachten Daten wie dem Masterfile MF und der (optional vorhandenen) Geldbörsenfunktion DF_Börse ist auf der erfindungsgemäßen Karte noch ein Verzeichnis bzw. eine Datenstruktur oder Dateistruktur DF_VAS vorgesehen. Diese dient zur Aufnahme von Zusatzfunktionalitäten, sogenannten Value Added Services VAS. Aufgrund dieser Zusatzfunktionalitäten, die Applikationen darstellen, die auch noch zu einem späteren Zeitpunkt, also nach der Herstellung der Karte, auf die Karte geladen werden können, ist die Karte hinsichtlich ihrer Funktionalitäten und der mit ihr durchführbaren Transaktionen flexibel und variabel. Der auf der erfindungsgemäßen Chipkarte vorgesehene sogenannte VAS-Container (DF_VAS) ermöglicht die Variabilität und Flexibilität der Chipkarte bezüglich ihrer Funktionen und löst daneben die auf der Chipkarte untergebrachten Applikationen sicherheitstechnisch von der Kartenplattform, so daß diese von der Kartenplattform unabhängig sind und eventuell sogar auf eine andere Karte übertragen werden können.

Die erfindungsgemäßen, neuen Zusatzfunktionalitäten (Value Added Services, VAS) werden realisiert durch Mikroprozessor-Chipkarten. Die Umsetzung dieser Zusatzfunktionalitäten wird durch den VAS-Container realisiert. Der VAS-Container auf der Mikroprozessor-Chipkarte bildet die Plattform zur Aufnahme der VAS-Applikationen. Die VAS-Applikationen sind die jeweiligen Realisierungen bestimmter Zusatzfunktionalitäten.

Im Fall der elektronischen Geldbörse wird das Bezahlen mit der Karte (Zahlungsfunktion) und die Nutzung einer VAS-Applikation (Zusatzfunktionalität) über getrennte Mechanismen abgewickelt; aus Sicht des Kartennutzers bzw. Karteninhabers kann dies aber als ein Vorgang erscheinen.

Die Mikroprozessor-Chipkarte wird erweitert um den VAS-Container, der verschiedene und unabhängige Applikationen aufnehmen kann. Er stellt Funktionen zum Aufbringen, Löschen und Übertragen dieser Applikationen zur Verfügung; diese können nur vom autorisierten Systembetreiber verwendet werden. Der VAS-Container ist daten- und sicherheitstechnisch unabhängig von anderen Systemkomponenten auf dem Mikroprozessor-Chip. Der VAS-Container ist vollständig durch sich selbst definiert und allein funktionsfähig. Für ihn ist eine unabhängige Sicherheitsarchitektur definiert, so daß VAS-Applikationen eigenständige Sicherheitsfunktionen verwenden. Die Sicherheitsarchitektur verwendet kartenspezifische Schlüssel, die nicht herstellerspezifisch und die unabhängig von Identifikationsmerkmalen der Kartenplattform sind.

Der VAS-Container verwendet auch Mechanismen zur Ableitung terminalspezifischer Schlüssel. Mit diesen kann der VAS-Container selbst aktiv die Echtheit von Terminals bzw. die von ihnen erzeugten Daten prüfen.

Im VAS-Container werden die VAS-Applikationen abgelegt, die über Mechanismen des VAS-Container Daten bereitstellen und dadurch die Steuerung der zugeordneten Schnittstellen bewerkstelligen. Der VAS-Container ermöglicht und steuert auch den sicheren Austausch von Daten für Interservices zwischen Partnern. Der VAS-Container erledigt aktiv die Kontrolle, d. h. die Echtheit und die Einmaligkeit, der übertragenen Datenwerte.

Ein Vorteil des VAS-Container gegenüber anderen Ansätzen von Multi-Applikations-Karten ist, daß dieses Konzept unabhängig von einer bestimmten Kartenplattform ist. Es bietet eine Sicherheitsarchitektur, die unabhängig von plattformspezifischen Sicherheitsmechanismen (wie Schlüssel, Identifizierungsdaten, PIN, Signaturverfahren) ist.

Ein weiterer Vorteil des Konzeptes VAS-Container ist, daß die Anzahl der verschiedenen Applikationen auf der Karte nicht fest vorgegeben ist durch Beschränkungen und Vorgaben zum Zeitpunkt der Kartenherstellung oder der Kartenausgabe; die aktuelle Kartenbelegung mit Anwendungen ist frei wählbar durch den Kartennutzer und ist nur durch den auf der jeweiligen Karte zur Verfügung stehenden Speicherplatz beschränkt. Die Anzahl der aktuell auf einer Karte geladenen VAS-Applikationen hängt vom tatsächlichen Gebrauch der Karte ab. Der Kartennutzer stellt individuell die VAS-Applikationen auf seiner Karte zusammen; dies beinhaltet auch die Änderung der Zusammenstellung zu einem späteren Zeitpunkt. Der VAS-Container ermöglicht eine multifunktionale Karte, bei der die Kartenfunktionalität während des Lebenszyklus der Karte variabel in Anzahl und Art der Anwendungen zusammengestellt und verwendet werden kann. Es können so auch Anwendungen, für die bislang einzelne, spezielle Karten notwendig waren, auf nur einer Karte abgelegt werden und zum Einsatz kommen. VAS-Applikationen können auch auf andere Karten übertragen werden. Damit können VAS-Applikationen den Lebenszyklus einer Karte überleben; sie begleiten den Kartennutzer während des Lebenszyklus der Anwendungen.

Die Mikroprozessor-Chipkarte mit Zusatzdiensten ist ein geeignetes Medium zur Verbreitung und zur Vermarktung von Dienstleistungen, bei denen auf geschützte Daten zugegriffen werden muß. Diese Mikroprozessor-Chipkarte kann als Zahlungsmittel, Recheneinheit und zur Wertaufbewahrung verwendet werden. Sie kann entsprechend dem Kundenwunsch vom Kunden selbst und nach Kartenausgabe flexibel mit Zusatzdiensten versehen und zu deren Steuerung herangezogen werden. Sie kontrolliert auch aktiv die Authentizität von beteiligten Terminals und stellt die Einmaligkeit und Echtheit von übergebenen Daten sicher.

Fig. 2 zeigt eine Systemübersicht. Darin dargestellt sind die Systemkomponenten.

Ein Systembetreiber (System Operator) stellt das System zur Verfügung. An Serviceterminals (ST) können VAS-Applikationen geladen, gelöscht und übertragen werden; weitere Operationen sind: VAS-Applikation auswählen, ansehen, interpretieren usw.

Die Anbieter von VAS-Applikationen, die sogenannten VAS-Provider, entwerfen eigene VAS-Applikationen nach den Rahmenbedingungen des Systembetreibers und, soweit möglich, nach eigenen Vorstellungen. Durch Abschluß mit einer digitalen Unterschrift werden die entsprechenden Terminalprogramme auf Echtheit nachprüfbar.

Fig. 3 zeigt den Datenfluß im System.

Die VAS-Applikationen werden über den Systembetreiber an den Terminals den Kartennutzern zur Verwendung bereitgestellt. VAS-Applikationen sind in den VAS-Container der erfindungsgemäßen Chipkarte zu laden, bevor daran teilgenommen werden kann.

An Händlerterminals (HT) wegen auf der Karte vorhandene VAS-Applikationen genutzt, indem VAS-Daten aufgebracht bzw. gelöscht werden.

Zur Realisierung der Mikroprozessor-Chipkarte mit VAS-Funktionalität wird neben anderen bestehenden Anwendungen (z. B. Zahlungsapplikationen wie die elektronische Geldbörse) der VAS-Container auf der Karte aufgebracht.

Der VAS-Container verwendet Funktionen, die ein Aufbringen, Löschen und Übertragen von VAS-Applikationen ermöglichen. Diese Verwaltungsfunktionen werden ausschließlich vom System Operator benutzt und sind in der Karte gegen Fremdbenutzung abgesichert. Der VAS-Container beinhaltet einen Transferspeicher, mit dem der Austausch von Daten zwischen VAS-Applikationen realisiert wird.

Zur Steuerung des Transferspeichers werden zwei Kommandos TRANSFER und TAKE eingesetzt. Das Kommando TRANSFER erzeugt einen Eintrag im Transferspeicher mit für die jeweilige Applikation spezifischen Daten. Neben den Nutzdaten zählen hierzu auch die zur Steuerung der Verarbeitung notwendigen Angaben wie Datum, Verfallsdatum und Identifikationsdaten. Mit dem Kommando TAKE werden Objekte aus dem Transferspeicher entnommen und als entnommen markiert. Je nach Anwendungsfall werden die Objekte dann als weiterhin gültig oder als ungültig markiert. Übertragene Daten werden durch den VAS-Container auf Echtheit und Einmaligkeit überprüft.

VAS-Applikationen verwenden zur Bereitstellung und Steuerung der Anwendungen VAS-Daten. Zugriff auf die VAS-

Daten wird durch die VAS-Applikation gesteuert, die sich dazu Mechanismen bedient, welche im VAS-Container allen Applikationen bereitgestellt sind. Ein VAS-Provider betreibt im VAS-Container eine oder mehrere VAS-Applikationen. Die Nutzung der VAS-Applikation definiert sich aus dem Aufbringen, Lesen und Verarbeiten von VAS-Daten.

Der VAS-Container unterstützt Interservices. Interservices erfordern Zugriff auf gemeinsame Daten, Übertragung von Leistungsansprüchen und die Abrechnung von Leistungen zwischen verschiedenen Partnern.

Im folgenden sollen die mit der erfindungsgemäßen Chipkarte möglichen Dienste bzw. Applikationen anhand von Beispielen aufgezeigt werden.

Die Beschreibung bezieht sich jeweils auf die Erscheinungsweise und Funktion nach dem Stand der Technik. Die Funktion soll zukünftig durch eine oder mehrere VAS-Applikationen nachgebildet werden.

Als erstes werden Intraservices vorgestellt.

Beispiel A: Kundenclub

Ein Kaufhaus betreibt einen Kundenclub. Der Kunde wird in diesem Club Mitglied und erhält mit diesem Status spezifische Clubleistungen, die Nichtmitglieder nicht erhalten können. Heute identifiziert sich ein Clubmitglied in der Clubumgebung durch einen Clubausweis. Der Clubausweis wird beim Eintritt erstellt, ist nicht übertragbar, und hat in der Regel eine Gültigkeitsdauer. Über den Clubausweis werden keine spezifischen Transaktionen abgewickelt, d. h. es besteht keine Kopplung mit dem Kundenumsatz. Damit grenzt sich der Clubausweis von Bonusprogrammen ab, in denen ein Zusammenhang Status/Umsatz besteht.

Analog: Großhändler Ausweis, Senator Card, Buchclub.

Ziel: Die Clubzugehörigkeit soll durch eine Applikation im VAS-Container nachgewiesen werden.

Beispiel B: Bonussystem

Ein Kunde erhält für jede Transaktion einen Bonusanspruch vergütet. Der Bonusanspruch wird kumuliert und kann an einem durch den Kunden definierten Zeitpunkt für eine geldwerte Leistung eingetauscht werden. Der Bonusanspruch gilt für einen definierten Zeitraum und kann anonym oder personenbezogen verwaltet werden. Der Bonusanspruch entsteht durch Umsatz oder Nutzungshäufigkeit.

Analog: Punktestand von Miles & More.

Ziel: Das Punktekonto soll durch eine Applikation im VAS-Container verwaltet werden.

Beispiel C: Rabatt

Der Kunde erhält Volumenrabatt nach Plan. Der Rabatt wird für jede einzelne Transaktion gewährt. Die Karte verwaltet keine Umsatzhistorie. Jede Transaktion ist in sich abgeschlossen.

Ziel: Der Rabattanspruch soll durch eine VAS-Applikation ausgewiesen werden.

Beispiel D: Ausweisfunktion

Eine Person kann durch Merkmale in der Karte gegenüber Dritten nachweisen, daß Berechtigung zu bestimmten Leistungen besteht. Die Zugehörigkeit der Person zum Ausweis muß bei jeder Transaktion nachgewiesen werden (Lichtbild, PIN, Biometrik). Die Echtheit des Ausweises wird als Sicherheitsmerkmal herangezogen.

Analog: Internet Zugang, Zugang Homebanking, Telefonkarte.

Ziel: Die Berechtigung soll durch eine VAS-Applikation nachgewiesen werden.

Beispiel E: Werteinheiten

Werteinheiten werden erworben und durch ein- oder mehrmalige Nutzung verbraucht. Pro Transaktion verringert sich der Wert um eine oder mehrere Einheiten. Die Nutzungsberechtigung ist übertragbar und kann Einschränkungen für die Nutzung beinhalten.

Analog: Einzelfahrschein, Streifenfahrschein, Abo-Konzertkarte, Squash-Zehnerkarte, Kino-Ticket, Telefoneinheiten.

Ziel: Durch eine VAS-Applikation soll die Abwicklung rationalisiert werden.

Beispiel F: Nutzungsabrechnung

Die Nutzung einer Leistung wird nach Zeit, Häufigkeit oder Menge registriert und nach Tarifplan abgerechnet. Im voraus ist nicht bekannt, in welchem Umfang die Leistung abgerufen wird.

Analog: Verzehrbon, Kurzzeitparkschein.

Ziel: Durch eine VAS-Applikation soll die Abrechnung nach Tarif ermöglicht werden. Die jeweils aktuell benötigten Abrechnungsdaten sollen im VAS-Container gespeichert sein.

Beispiel G: Merkzettel (Mobiler Datenspeicher)

Diese Applikation ermöglicht die Übergabe von Daten des Karteninhabers an den VAS-Provider. Dadurch werden Abläufe automatisiert, die derzeit noch manuell erfolgen. Aus diesen Daten läßt sich kein geldwerter Gegenwert ableiten.

Analog: Ausfüllen von Lottoscheinen, Einkaufszettel, Kassenzettel, Telefonregister.

Ziel: Ein VAS-Provider soll die Daten der Karte entnehmen und so den entsprechenden Dienst direkt erbringen können

(z. B. Telefonverbindung herstellen, gewünschte Waren zusammenstellen, elektronisch den Lottoschein ausfüllen und erfassen). Die Daten können sowohl kurzfristig als auch langfristig in der Karte gespeichert werden.

Nach diesen Beispielen für Intraservices sollen nun einige Beispiele für Interservices folgen.

Ein Interservice entsteht immer dann, wenn an einem Dienst mehrere VAS-Provider beteiligt sind. Dies ist untrennbar damit verbunden, daß Daten den Bereich eines VAS-Provider verlassen. Heute geschieht dies über Bescheinigungen auf Papier.

Beispiel A: Fahrpreiserstattung

Ein Kaufhaus erstattet einem Kunden den Fahrschein eines ÖPNV-Unternehmens für die Fahrt zum Kaufhaus. Der Kunde muß dem Kaufhaus den Einzelfahrschein vorlegen. Das Kaufhaus vermerkt auf dem Fahrschein, daß er erstattet wurde. Eventuell bekommt das Kaufhaus seinerseits einen Teil der Erstattung an den Kunden vom Verkehrsunternehmen zurück.

Ziel: Der Erstattungsvorgang soll elektronisch abgewickelt werden.

Beispiel B: Fahrgutschein

Ein Kaufhaus erstattet dem Kunden beim Warenkauf die Kosten für eine Heimfahrt, indem ein Gutschein ausgestellt wird. Der Kunde erhält an einem Fahrkartenschalter ein Ticket des ÖPNV bzw. zahlt einen geringeren Preis für das Ticket. Der Verkehrsbetrieb rechnet den Gutschein mit dem Kaufhaus ab.

Ziel: Abbildung der Mechanismen durch VAS-Applikationen mit elektronischer Abrechnung zwischen Handel und ÖPNV.

Beispiel C: Kundenparkplatz

Ein Kaufhaus erstattet dem Kunden einen Teil der Parkgebühren bei Benutzung eines bestimmten Parkhauses. Das Parkhaus wird durch ein unabhängiges Unternehmen betrieben und erhält vom Kaufhaus eine geldwerte Vergütung für jede Kundengutschrift.

Ziel: Abbildung der Mechanismen durch VAS-Applikationen mit elektronischer Abrechnung zwischen Handel und Parkhaus.

Beispiel D: Multilaterales Bonusprogramm

Eine Gruppe von Handelsunternehmen und Dienstleistern einigt sich auf ein gemeinsames Bonusprogramm.

Ziel: Jeder Partner kann Bonuspunkte auf einem gemeinsamen Konto auf der Karte gutschreiben oder vergüten. Die Abrechnung der Leistungen zwischen den Partnern erfolgt durch das Hintergrundsystem.

Beispiel E: Anerkennung von Bonuspunkten zwischen Dienstleistern

Jeder Dienstleister betreibt ein eigenes Bonusprogramm, hat aber Absprachen mit anderen, gesammelte Punkte anzuerkennen. Bekannte Beispiele sind Absprachen zwischen Autovermietern und Fluggesellschaften zum Sammeln von "Meilen".

Ziel: Unterstützung der Übertragung von Bonuspunkten durch die Karte.

Jeder VAS-Provider soll zunächst seine Punkte für sich sammeln, ohne daß ein anderer eingreifen kann. Für die Übertragung sollen sichere Mechanismen bereitgestellt werden.

Beispiel F: Nachttaxi

Durch einen Fahrausweis des ÖPNV wird gleichzeitig die Berechtigung zur Fahrt im Sammeltaxi (z. B. nach 22.00 Uhr) erworben. Das Taxiunternehmen muß zur Abrechnung nachweisen, daß der Fahrausweis vorgelegen hat. Die Inanspruchnahme des Taxis wird auf der Karte vermerkt, um Mißbrauch zu unterbinden.

Ziel: Die VAS-Applikation soll die Inanspruchnahme, Kontrolle und Verrechnung dieses Dienstes ermöglichen.

Im folgenden soll der Aufbau der erfindungsgemäßen Chipkarte näher beschrieben werden.

Die Mikroprozessor-Chipkarte mit VAS-Funktionalität enthält insbesondere den VAS-Container.

Der VAS-Container ist eine eigenständige Anwendung und existiert entweder alleine oder auch parallel neben anderen Applikationen auf der zugrundeliegenden Kartenplattform. Der VAS-Container ist vollständig durch sich selbst definiert und allein funktionsfähig. Er kann ohne die in der Regel auf der gleichen Karte vorhandene Zahlungsfunktion betrieben werden. Insbesondere wird für den VAS-Container eine unabhängige Sicherheitsarchitektur definiert, so daß VAS-Applikationen eigenständige Sicherheitsfunktionen verwenden können.

Teil der Value Added Services sind Transaktionen durch den Kunden, die an Terminals der VAS-Provider durchgeführt werden. Der VAS-Provider hat ein Interesse, diese Transaktionen zu verfolgen, sei es zum Zweck der Systemüberwachung oder sei es zur Sammlung von statistischen und anderen Daten. Zur Optimierung und Vereinheitlichung der Datenstrukturen in der Karte wird von der Verwendung applikationsspezifischer Identifikationen abgeraten und die Nutzung einer systemweit eindeutigen VAS-Container ID empfohlen. Diese Nummer kann vom VAS-Provider zur Realisierung der o.g. Funktionen benutzt werden, und sie befreit ihn von der Bürde, eigene Nummernkreise zu verwalten.

Die Sicherheitsarchitektur des VAS-Containers verwendet diese systemweit eindeutige ID zur Ableitung kartenspezifischer Schlüssel. Die Verwendung der kartenspezifischen Nummer wäre im Prinzip möglich, wird aber vorzugsweise nicht verwendet, da, falls der VAS-Container auf unterschiedlichen Plattformen implementiert wird, diese unter Umstän-

den kollidierende Nummernkreise verwenden.

Die VAS-Container ID wird vom System Operator vergeben und durch den Kartenissuer bei der Erzeugung des VAS-Containers eingebracht. Sie wird mit dem Löschen des VAS-Containers vernichtet und damit aus dem System entfernt. Ein VAS-Container gilt als gelöscht, wenn er keine VAS-Applikationen enthält und die VAS-Container ID entfernt ist.

Bei einer Gesamtübertragung des VAS-Containers von einer alten in eine neue Karte wird die VAS-Container ID zusammen mit den VAS-Applikationen transferiert. Dieser Transfer entspricht einem Verschieben des VAS-Containers aus der alten in die neue Karte. Die alte Karte enthält nach diesem Vorgang keinen VAS-Container mehr. Der in der neuen Karte vorhandene VAS-Container wird bei dieser Operation überschrieben und damit gelöscht. Da die VAS-Container ID von der alten auf die neue Karte übertragen wird, ist in den Hintergrundsystemen der VAS-Provider keine Umwandlung der Bezugsnummern erforderlich.

Einzelne VAS-Applikationen können nur unter der Kontrolle des jeweiligen VAS-Providers zwischen zwei verschiedenen VAS-Containern verschoben werden. Bei dieser Funktion ändert sich die Zuordnung zwischen VAS-Container ID und VAS-Applikation, was u. U. im Hintergrundsystem des VAS-Providers vermerkt werden muß.

Der VAS-Container enthält keine personenbezogenen Merkmale. VAS-Applikationen können personenbezogene Daten enthalten, der Umfang sollte jedoch aus Datenschutzgründen und zur besseren Speicherausnutzung auf ein Minimum beschränkt werden. VAS-Applikationen sollen, falls erforderlich, personenbezogene Daten im Hintergrundsystem speichern und die Zuordnung zur Karte über die VAS-Container ID herstellen.

Ein wesentliches Merkmal des VAS-Containers ist die Unterstützung von Interservices. Interservices erfordern Zugriff auf gemeinsame Daten, Übertragung von Leistungsansprüchen und die Abrechnung von Leistungen zwischen verschiedenen Partnern. Der VAS-Container muß dieses ermöglichen und trotzdem die Sicherheit der Applikationen im Intraservice gewährleisten.

Sogenannte Intraservice VAS-Applikationen sind Applikationen, die unter der exklusiven Kontrolle des VAS-Providers betrieben werden. Der VAS-Provider definiert die Sicherheit seiner Applikation unabhängig von einer externen Stelle. Ohne Weitergabe der Schlüssel kann keine externe Partei VAS-Daten verändern.

Zur effizienten Abwicklung von Interservices müssen die Partner auf gemeinsame Daten zugreifen können. Der gemeinsame Zugriff auf die Daten ist über abgestufte Sicherheitsmechanismen realisiert.

Die erste Stufe des gemeinsamen Zugriffs erfolgt ausschließlich über öffentliche Transferfelder. VAS-Provider können über diese Felder Daten austauschen, ohne gegenseitig die Applikationen und Schlüssel kennen zu müssen. Lediglich zum Schreiben von Daten in die Transferfelder müssen die Terminals über geeignete Schlüssel verfügen, nicht aber zum Lesen. Lesen darf jeder ohne Einschränkung. Der Datenaustausch kann in beiden Richtungen zwischen VAS-Provider und Partner erfolgen, wenn jeder über seinen Schlüssel zum Schreiben verfügt. Die Transferdaten können aus einer Intraservice VAS-Applikation des VAS-Provider erzeugt werden oder umgekehrt kann der VAS-Provider Daten aus dem Transferfeld in seine Intraservice VAS-Applikation hereinnehmen.

Die zweite Stufe ist das Entwerten von Werteinheiten, die durch VAS-Daten verkörpert werden. Ein VAS-Provider bringt Werteinheiten in die VAS-Daten mit einem speziellen Schlüssel zum Schreiben ein. Die Werteinheiten können durch Interservice Partner abgenutzt werden, die über einen Schlüssel zum Entwerten verfügen, den sie vom gesamtverantwortlichen VAS-Provider bekommen. In dieser Stufe greifen Partner mit unterschiedlichen Rechten auf nicht-öffentliche VAS-Daten zu.

Die dritte Stufe ist der direkte schreibende Zugriff auf die VAS-Daten durch alle beteiligten Interservice Partner. Diese Methode erfordert ein entsprechendes Maß an Vertrauen zwischen den Parteien, da uneingeschränkt VAS-Daten verändert werden können.

Das Transferfeld dient als Koppelglied zwischen VAS-Applikationen. Daten im Transferfeld werden von den VAS-Applikationen im VAS-Container erzeugt. Daten können auch direkt im Transferfeld angelegt werden, wenn der Erzeuger keine eigene VAS-Applikation im VAS-Container betreibt.

Das Transferfeld steht prinzipiell allen Applikationen zur Verfügung, schreiben darf aber nur wer über eine Berechtigung (Schlüssel) dafür verfügt. Nur nach den Regeln, d. h. den Rules und Regulations R&R, dazu berechtigte Empfänger dürfen Daten aus dem Transferfeld entnehmen. Der Empfänger prüft das Vorhandensein von für ihn bestimmten Transferdaten und entnimmt sie zur Verarbeitung im eigenen System.

Um die Manipulation von Transferdaten im öffentlichen Austausch zu verhindern, bringt der Erzeuger ein Echtheitsmerkmal und der VAS-Container eine Sequenznummer ein. Mit diesen Elementen wird die Einmaligkeit einer Transfernachricht sichergestellt und der Ursprung der Daten nachgewiesen.

Bei Bedarf wird der Empfänger der Transferdaten vom Erzeuger mit der Möglichkeit ausgerüstet, eine Echtheitsprüfung durchzuführen. Wird dies nicht gewünscht, kann die Akzeptanz der Daten auch nach Treu und Glauben erfolgen; das Echtheitsmerkmal wird dann u. U. erst bei einer Abrechnung im Hintergrundsystem des erzeugenden VAS-Provider geprüft.

Die Daten können aus dem Transferfeld nur einmalig mit Erhalt eines Echtheitsmerkmals entnommen werden. Bei der Entnahme werden die Daten markiert und verbleiben (als Kopie) im Transferfeld. Damit kann auch nach der Entnahme der Daten für einen bestimmten Zeitraum der Transfervorgang nachgewiesen werden.

Daten im Transferfeld tragen ein Verfallsdatum. Verfallene Daten können bei Bedarf von beliebigen Applikationen überschrieben werden. Die Daten im Transferfeld können bei der Entnahme markiert werden, so daß sie zum sofortigen Überschreiben freigegeben sind. Sollte sich der Transferspeicher dennoch vollständig füllen, bevor ein Transferdatensatz verfällt, so muß der Karteninhaber am Serviceterminal nicht mehr benötigte Einträge löschen.

Für die Modellierung von VAS-Applikationen werden 3 Operationen definiert. Diese Operationen wirken auf die VAS-Daten. Laden und Löschen von Applikationen sind Funktionen des VAS-Containers und werden in den folgenden Absätzen nicht betrachtet.

Erwerben Allgemein das Erwerben eines Leistungsanspruchs und die Ablage eines Nachweises in den VAS-Daten einer VAS-Applikation.

Entwerten Allgemein das Einlösen des Anspruchs ohne, mit vollständigem oder mit teilweisem Verbrauch von VAS-

Daten einer Applikation. Der Vorgang erzeugt eine elektronische Quittung, die im Transferfeld abgelegt wird. Diese Quittung trägt ein sinnvolles Verfallsdatum, zu dem sie aus dem Transferspeicher gelöscht werden kann.

Entnehmen Allgemein die Entnahme der elektronischen Quittung aus einem Transferfeld zur weiteren Verarbeitung (z. B. Hintergrundsystem). Eine Kopie der Quittung verbleibt und dient als Nachweis des "Entwertens".

"Erwerben" von VAS-Daten kann nur durch den jeweiligen VAS-Provider erfolgen. "Entwerten" von VAS-Daten kann nur durch den VAS-Provider erfolgen, der sie mit "Erwerben" erzeugte oder durch einen Interservice Partner, den der VAS-Provider dazu ermächtigt. "Entnehmen" kann auch durch jeden anderen VAS-Provider erfolgen. Bei einem Interservice ohne gleichberechtigten Zugriff auf die VAS-Daten löst der Partner den Zustandsübergang "Entnehmen" aus. Die so gewonnene elektronische Quittung kann dann über den System Operator und das Hintergrundsystem des VAS-Providers abgerechnet werden. "Erwerben" und "Entwerten" kann in einem Schritt erfolgen.

VAS-Applikationen mit gemeinsamen Merkmalen lassen sich in Klassen unterteilen. Diese Klassen bilden die Basis für Datenstrukturen im VAS-Container. Der VAS-Provider wählt bei der Implementation seiner Anwendung eine Applikationsklasse aus.

Es werden dabei folgende Applikationsklassen definiert:

- Punktespeicher
- Ticket
- Ausweis
- Gutschein
- Datenspeicher.

Punktespeicher bezeichnet eine Applikationsklasse, in der ein Konto von Punktwerten verwaltet wird. Auf das Punktekonto können Werte aufgebucht und abgezogen werden. Das Aufsuchen von Werten erfolgt durch den VAS-Provider, indem der Speicher mit dem neuen Kontostand beschrieben wird. Das Abbuchen von Werten erfolgt durch die "Entwerten"-Operation, wobei als Beleg eine elektronische Quittung im Transferspeicher abgelegt wird. Der Zugriff auf die beiden Funktionen ist durch zwei unterschiedliche Zugriffsschlüssel realisiert.

Ticket bezeichnet eine Applikationsklasse, in der ein Wertfeld existiert, welches einmalig oder mehrmalig entwertet und damit verbraucht wird. Das Aufbuchen von Werten in der Applikationsklasse Ticket erfolgt einmalig.

Ausweis bezeichnet eine Applikationsklasse, bei der die VAS-Daten den Nachweis einer Berechtigung beinhalten. Der Nachweis wird in der Regel nicht durch Nutzung verbraucht; er wird allerdings nach einem definierten Kriterium, z. B. nach zeitlichem Verfall, ungültig. Je nach Ausprägung der Applikation wird die Echtheit des Ausweises (z. B. Anwohnerparkschein) oder die Vorlage eines Ausweises dokumentiert (z. B. Sammeltaxifahrt mit Monatsfahrschein: Erzeugen einer elektronischen Quittung durch die Karte. Die Quittung wird vom Taxiunternehmen beim ÖPNV eingereicht und anteilig verrechnet). Optional kann die Nutzung des Ausweises durch elektronische Quittungen im Transferspeicher der Karte dokumentiert werden.

Gutschein bezeichnet eine Applikationsklasse, bei der Leistungsansprüche (in der Regel kurzfristig) in der Karte zwischengespeichert werden. Diese elektronischen Gutscheine werden ausschließlich im Transferspeicher des VAS-Containers gespeichert. Die den Gutschein verwertende Applikation ist in der Regel eine andere als die erzeugende Anwendung. Die Entnahme des Gutscheins aus dem Transferspeicher ist nur einmalig möglich und wird durch die Karte dokumentiert. Ein vom VAS-Container erzeugtes Echtheitsmerkmal kann bei der Abrechnung im Hintergrundsystem benutzt werden.

Datenspeicher bezeichnet eine Applikationsklasse, bei der ein VAS-Provider Daten im VAS-Container speichert, die für den Kunden einen zusätzlichen Service bieten (z. B. Letztes Menü in Fast-Food Restaurant, Letzte Lottozahlen, Service Präferenzen, Vorschlagslisten). Die Daten sind nur zur Information und stellen keinen Leistungsanspruch gegen den VAS-Provider dar. Der Zugriff auf die Daten wird vom VAS-Provider gesteuert.

Jede Applikationsklasse zeichnet sich durch einen typischen Nutzungszyklus aus. Die folgende Tabelle zeigt, in welcher Frequenz die drei oben definierten Operationen auf die VAS-Daten der Applikationsklasse angewendet werden (Zur Beachtung: "Erwerben" bedeutet nicht "Applikation laden" und "Entnehmen" bedeutet nicht "Applikation löschen").

Tabelle 1

Nutzungszyklus

	Punkte-speicher	Ticket	Ausweis	Gutschein	Daten-speicher
Erwerben	Mehrfach	Einfach	Einfach	Einfach	Mehrfach ***
Entwerten	Mehrfach	Mehrfach	Mehrfach	Einfach	Nie
Entnehmen	Mehrfach	Mehrfach	Mehrfach	Einfach	Nie

*** In der Applikationsklasse „Datenspeicher" wird nicht ein Anspruch erworben, sondern die Applikation trägt lediglich Daten in die Applikation ein.

Fig. 4 veranschaulicht die oben aufgeführten Applikationsklassen und Operationen durch ein Transaktionsmodell.

Im folgenden soll auf die Sicherheitsarchitektur der erfindungsgemäßen Chipkarte näher eingegangen werden. Zur Gewährleistung der Sicherheit werden die folgenden Schlüssel definiert:

Tabelle 2

Schlüsselübersicht

Name	Ort	Inhaber	Beschreibung
K_{SO}	VAS-Container	System Operator	Schlüssel für Verwaltung des VAS-Containers
K_{AUT}	VAS-Container	System Operator	Schlüssel für Authentifizierung des VAS-Containers
$K_{SIG\ VASC}$	VAS-Container	System Operator	Schlüssel zum Signieren der Transaktionsdaten
KGK_{DEC}	VAS-Container	System Operator	Schlüssel für die Ableitung des $KGK_{DEC.PIX}$
K_{LVASP}	VAS-Applikation	VAS-Provider	Schlüssel für den Schreibzugriff auf VAS-Daten
K_{RVASP}	VAS-Applikation	VAS-Provider	Schlüssel für den Lesezugriff auf VAS-Daten
$KGK_{DEC.PIX}$	VAS-Provider	VAS-Provider	Schlüssel für die Ableitung des K_{DEC}
K_{DEC}	Händler-terminal	VAS-Provider	Schlüssel für Authentifizierung des Händlerterminals

Die Sicherheitsarchitektur basiert auf dem Lebenszyklus von VAS-Container bzw. VAS-Applikationen und ist nach der Verantwortlichkeit der beteiligten Instanzen geschichtet. Eine erläuterende schematische Darstellung ist in **Fig.** 5 gezeigt.

Nachfolgend einige Erläuterungen zur Struktur des VAS-Containers sowie der darauf aufgebrachten Applikationen.

Der VAS-Container sowie VAS-spezifische Ergänzungskommandos werden entweder vom Issuer zusammen mit anderen Nicht-VAS-Applikationen auf die Karte gebracht oder spätestens an einem Serviceterminal durch einen autorisierten VAS-Provider auf einer bestehenden Kartenplattform integriert.

Für die zweite Möglichkeit kann der folgende Mechanismus Verwendung finden: Der System Operator verabredet mit dem Issuer einen temporären Schlüssel K_{SO}*. Der Issuer öffnet die Karte mit seinem nur ihm bekannten Schlüssel, legt die Dateien des VAS-Container an und schreibt insbesondere den K_{SO}* in das DF_VAS. Der System Operator kann dann zu einem späteren Zeitpunkt (z. B. die Karte kommt zum ersten Mal mit einem Serviceterminal in Berührung) den Schlüssel K_{SO}* durch seinen eigenen Schlüssel K_{SO} ersetzen, den dann nur er alleine kennt. Der System Operator kann nun weitere Daten, wie z. B. den KGK_{DEC}, selbst eintragen oder im Falle einer Plattform mit dynamischer Speicherverwaltung Dateien für VAS-Applikationen selbst erzeugen bzw. löschen. Damit ist sichergestellt, daß der Issuer nach der ersten Benutzung einer VAS-Karte keinen Zugriff mehr auf den VAS-Container hat und der System Operator lediglich

Zugriff sauf den VAS-Container. Da es mit anderen Applikationen keine gemeinsamen Datenstrukturen und keinerlei Datenaustausch gibt, ist die Sicherheitsarchitektur des VAS-Containers somit unabhängig von anderen auf der Kartenplattform vorhandenen Applikationen.

In einer speziellen Ausführungsform der Erfindung soll jedoch von der ersten Möglichkeit Gebrauch gemacht werden: Der Issuer soll im Auftrag des System Operator den VAS-Container inklusive aller Schlüssel auf die VAS-Karten aufbringen.

Der VAS-Container besteht aus einer vorgegebenen Dateistruktur, vorgegebenen Zugriffsbedingungen (ACs) und einigen globalen Daten. Die globalen Daten enthalten u. a. den Schlüssel K_{SO} zum Laden bzw. Löschen von VAS-Applikationen. Über diesen Schlüssel, den nur der System Operator kennt, wird sichergestellt, daß nur zugelassene VAS-Applikationen geladen werden können. Die Karte verlangt dazu eine externe Authentifizierung des System Operators mit K_{SO}.

Wenn ein Karteninhaber an einem Serviceterminal eine VAS-Applikation laden möchte, beauftragt der zuständige VAS-Provider den System Operator damit, dies für ihn zu tun. Beim Laden der VAS-Applikation in den VAS-Container übergibt der VAS-Provider dem System Operator die Schlüssel K_{LVASP} und K_{RVASP}, die dieser mit in die Applikation einträgt. Durch den Schlüssel K_{LVASP} kann der VAS-Provider Daten der Applikation vor schreibendem Zugriff und zusätzlich interne Daten vor lesendem Zugriff schützen. Dazu verlangt die Karte eine externe Authentifizierung des VAS-Provider basierend auf K_{LVASP}; d. h. die Karte prüft aktiv die Echtheit des Terminals. Nach erfolgreicher Ausführung dieser Funktion wird einem Terminal der schreibende Zugriff auf eine VAS-Applikation und der lesende Zugriff auf interne VAS-Daten der Applikation erlaubt. Eine interne Authentifizierung, d. h. die Prüfung der VAS-Applikation (und damit der Karte) auf Echtheit durch das Händlerterminal, kann optional erfolgen. Bei der Nutzung der VAS-Applikation durch den Karteninhaber können an beliebigen Terminals, die über den Schlüssel K_{LVASP} verfügen, diese internen VAS-Daten in die Applikation geschrieben bzw. verändert werden. Die Funktion "Erwerben" stützt sich deshalb auf einen UPDATE RECORD Befehl, dem eine externe Authentifizierung mit dem Schlüssel K_{LVASP} vorausgehen muß.

Der lesende Zugriff auf alle nicht internen Daten der VAS-Applikation ist nur erlaubt, wenn eine vorherige externe Authentifizierung durch K_{RVASP} oder durch K_{SO} oder durch die korrekte Eingabe einer PIN oder eines Paßwortes durch den System Operator erfolgt. Der PIN-/Paßwort-geschützte Zugriff wird vorgesehen, um Daten für den Karteninhaber an Terminals oder am Wallet einsehbar zu machen. Der Karteninhaber hat an einem Serviceterminal die Wahl, für alle Applikationen gleichzeitig einen PIN-/Paßwort-Schutz für das Lesen der Wert- bzw. Statusdaten zu aktivieren bzw. zu deaktivieren.

Zu den globalen Daten des VAS-Container gehört ein Schlüssel K_{SIG_VASC} zum Signieren von entnommenen Daten aus dem Transferfeld. Anhand der Signatur kann die Unversehrtheit dieser Transaktionsdaten nachgeprüft werden, wenn sie manipulationsgesichert zwischen Interservice Partnern zur gegenseitigen Verrechnung übertragen werden müssen. Zusätzlich zu einer optional durch die Interservice Partner eingebrachten Signatur wird für Datensätze, die die Karte mit der Operation "Entnehmen" ausgibt, eine Signatur basierend auf K_{SIG_VASC} und einem vom VAS-Container verwalteten Transaktionszähler hinzugefügt. Da zwar Lesen des Transferfeldes jedem erlaubt ist, jedoch die Signatur der Karte mit K_{SIG_VASC} nur beim Aufruf der Operation "Entnehmen" erzeugt wird (und dies kann nur einmalig geschehen), wird die unzulässige doppelte Abrechnung von Gutscheinen erkannt. Jeder Interservice Partner kann sich vom System Operator die Echtheit und Einmaligkeit eines entnommenen Gutscheins bestätigen lassen. Außerdem kann vom System Operator durch Prüfung der Signatur die Echtheit des VAS-Containers nachgewiesen werden.

Sollte eine Kartenplattform asymmetrische Schlüsselverfahren unterstützen, kann als K_{SIG_VASC} auch ein privater (geheimer) Schlüssel innerhalb der Karte zum Signieren herangezogen oder ein solcher Schlüssel aus einem privaten Key Generating Key abgeleitet werden. Die VAS-Provider könnten in diesem Fall öffentliche (nicht geheime) Schlüssel zur eigenen Prüfung der Signatur heranziehen, ohne den System Operator konsultieren zu müssen.

Zu den Daten des VAS-Container gehört ein globaler Key Generating Key KGK_{DEC}, der für alle Terminals aller VAS-Provider zur Berechtigungsprüfung der Operation "Entwerten" den Zugriffsschlüssel K_{DEC} erzeugen kann (mehr zur Schlüsselableitung und Prüfung folgt später). An das Entwerten von geldwerten Daten werden nicht die gleichen Sicherheitsmaßstäbe wie an das Laden bzw. Erwerben eines Anspruchs gestellt. Es genügt hier, anstelle eines applikationseigenen Schlüssels einen globalen Schlüssel zu verwenden. Dieser globale Schlüssel wird durch Ableitung zunächst in einen Applikationsschlüssel überführt und anschließend durch erneutes Ableiten zum Terminalschlüssel. Dem VAS-Provider wird nur die applikationsspezifische Ableitung des globalen Schlüssels zur Erstellung eigener Terminalschlüssel übermittelt. Dies wird nachfolgend kurz geschildert:

Wir bezeichnen mit mac(k,d) die Berechnung eines Message Authentication Code für eine Nachricht d und einen DES-Schlüssel k mittels DES. Falls die Nachricht nicht länger als 8 Bytes ist, entspricht dies der Verschlüsselung selbst (ICV=0 vorausgesetzt). Wir bezeichnen mit macp(k,d) die Berechnung von mac(k, d) mit anschließender Angleichung der Paritätsbits. Das Ergebnis k' = macp(k,d) ist wieder ein gültiger DES-Schlüssel.

Die Berechnung des applikationsspezifischen Terminalschlüssels geschieht dann wie folgt:

1. Jede Karte speichert einen Schlüssel

KGK_{DEC},

der für alle Karten gleich ist und der das Geheimnis des System Operators ist. Der System Operator veranlaßt, daß dieser Schlüssel in die Karte personalisiert wird.
Aus diesem Schlüssel können sämtliche VAS-Applikations- und Terminalschlüssel abgeleitet werden.
2. Ein VAS-Provider möchte eine VAS-Applikation A mit $AID_A = RID_{VAS} \cdot PIX_A$ einführen. Nun berechnet der System Operator aus dem Schlüssel KGK_{DEC} und der PIX den applikationsspezifischen Schlüssel
$KGK_{DEC,PIX} = macp(KGK_{DEC}, PIX_A)$
und übergibt diesen Schlüssel dem VAS-Provider.

3. Dieser VAS-Provider leitet aus $KGK_{DEC,PIX}$ für alle Terminals, die das TRANSFER-Kommando auf die VAS-Applikation A anwenden sollen, mit Hilfe ihrer Terminal ID ihre spezifischen Schlüssel ab:
K_{DEC} = macp($KGK_{DEC,PIX}$, Terminal ID) = macp(macp(KGK_{DEC}, PIX_A), Terminal ID).
Der VAS-Provider speichert diese Schlüssel in seinen Terminals. Sofern der VAS-Provider nicht selbst die Terminals betreibt, generiert er die Schlüssel und verteilt sie an die Betreiber der Terminals.
4. Die VAS-Karte führt das TRANSFER Kommando nur durch, wenn das Kommando ein vom Terminal erzeugtes Kryptogramm C enthält, das über bestimmte Daten data der Nachricht im Transferspeicher gebildet wird. Die Karte selbst kennt KGK_{DEC} und bekommt von einem Terminal neben den Nutzdaten data und dem Kryptogramm C die Terminal ID sowie die PIX geschickt (bzw. die Karte kennt die PIX selbst, siehe hierzu auch die Beschreibung des Kommandos TRANSFER). Nun kann die Karte das Kryptogramm C' über die Nutzdaten selbst berechnen:
mac(macp(macp(KGK_{DEC},PIX), Terminal ID),data) = = mac(macp($KGK_{DEC,PIX}$, Terminal ID),data) = = mac(K_{DEC},data) = = C'.

Die Karte vergleicht nun das selbst berechnete Kryptogramm C' mit dem vom Terminal berechneten Kryptogramm C. Sind sie unterschiedlich, erfolgt ein Abbruch der Transaktion mit einer Fehlermeldung. Im Falle der Übereinstimmung wird die VAS-Karte das TRANSFER-Kommando ausführen.

Die unterschiedlichen Sicherheitsmaßstäbe werden den unterschiedlichen Bauweisen von Serviceterminals bzw. Händlerterminals (zum Laden bzw. Erwerben) und einfachen Händlerterminals (zum Entwerten) gerecht. Ein Terminal muß sich zur Ausführung der "Entwerten" Operation gegenüber der Karte durch Kenntnis eines Schlüssels K_{DEC} authentisieren. Bei Kompromittierung des Schlüssels K_{DEC} kann der Angreifer lediglich die Funktionen dieses einen Terminals durchführen. Dieser Vorgang wird jedoch in der Karte protokolliert. Die Dokumentation enthält eine eindeutige Sequenznummer, die Entwertung und die Terminal ID. Die VAS-Applikationen nutzen die Sicherheitsdienste des VAS-Containers, um den Zugriff auf VAS-Daten zu reglementieren. Die Ausführung von VAS spezifischen Funktionen wird durch die Definition von Zugriffsrechten auf berechtigte Parteien beschränkt.

Im allgemeinen muß es für jede VAS-Applikation öffentlich zugängliche Datenbereiche (z. B. zum Auslesen von Werten mit einem Wallet) und private Datenbereiche des verantwortlichen VAS-Provider geben, die er vor dem Zugriff durch andere schützen kann (z. B. interne Verwaltungsdaten).

Da Karten nach ISO 7816-4 für einzelne Records von Elementary Files (EF) keinen differenzierten Zugriffsschutz bieten, müssen mehrere Dateien, die jeweils einen Record enthalten, zur Darstellung unterschiedlicher Sichten auf eine VAS-Applikation verwendet werden.

So läßt sich für die Applikationsklassen Punktespeicher, Ticket, Ausweis und Datenspeicher differenzierter Zugriffsschutz durch die Aufteilung der VAS-Daten in vier Elementary Files realisieren, wie in **Fig.** 6 dargestellt.

Die vier Elementary Files enthalten folgende Daten bzw. werden folgendermaßen geschützt:

Tabelle 3

Übersicht Dateien

Feld	Inhalt	Typische Nutzung	Zugriffsschutz
EF_KEY	Enthält die Schlüssel K_{LVASP}, K_{RVASP} die nur der VAS-Provider kennt (Anmerkung: Pro VAS-Applikation und optional pro Karte vergibt der VAS-Provider jeweils ein Schlüsselpaar)	Ein VAS-Provider muß sich mit K_{LVASP} authentisieren, bevor er VAS-Daten schreiben darf in EF_INFO, EF_INTERNAL und EF_VALUE bzw. bevor er Daten aus EF_INTERNAL lesen darf. Ein Terminal muß sich mit K_{RVASP} authentisieren, bevor es VAS-Daten aus EF_INFO, EF_VALUE lesen darf.	
EF_INFO	Nicht interne Informationen über die VAS-Applikation	• Ticketinformationen • Bonusprogramm Information • Ausweisinformationen • Parkschein Informationen	Die Inhalte dieser Dateneinheit können nur vom VAS-Provider geschrieben (externe Authentifizierung durch Kenntnis von K_{LVASP}) werden. Lesen soll nur an Terminals, die über K_{RVASP} oder K_{SO} verfügen, möglich sein, oder der Karteninhaber gibt einen korrekten PIN ein. (Der PIN-Schutz kann von Karteninhaber deaktiviert werden.)
EF_INTERNAL	Interne Daten der VAS-Applikation	Interne Zähler, Kontostände, Steuerinformationen, zusätzliche Keys für: • Bonusprogramme • Rabattstaffelungen • Verdeckte Ausweismerkmale • Codes	Die Inhalte dieser Dateneinheit können nur vom VAS-Provider beschrieben und gelesen werden. Der Zugriffsschutz basiert auf externer Authentifizierung durch Kenntnis von K_{LVASP}.
EF_VALUE	Werteinheiten der VAS-Applikation	Diese Dateneinheit enthält buchbare Werte einer VAS-Provider Applikation. • Kontostand • Restwert Fahrschein • Guthaben • Nutzungszähler	Nur der VAS-Provider kann diese Werte explizit schreiben (externe Authentifizierung durch Kenntnis von K_{LVASP}). Implizit kann der Wert durch den Befehl „Entwertung" von einem Partner verringert werden, der zur Ausführung der Funktion durch den VAS-Provider berechtigt wurde (durch Signatur der Operation Entwertung mit K_{DEC}). Lesen wie bei EF_INFO.

Diese Struktur von Elementary Files (EF) unterstützt die gleichen differenzierten Zugriffsrechte für alle Applikationsklassen.

Indem nun gleichartige Applikationsklassen zu jeweils einer Implementierungsklasse, die Anzahl und Größe der Elementary Files festlegt, zusammengefaßt werden, läßt sich ein Speicherverschnitt durch unterschiedlichen Platzbedarf für Applikationen minimieren. Die Applikationsklassen Punktespeicher und Ticket benötigen die gleichen Resourcen EF_KEY, EF_INFO, EF_INTERNAL, EF_VALUE und werden daher zusammengefaßt zur Implementierungsklasse "DF_PT". Für die Applikationsklassen Ausweis und Datenspeicher genügen die Elementary Files EF_KEY und EF_INFO und werden daher zur Implementierungsklasse "DF_AD" zusammengefaßt. Anzahlmäßig betrachtet: Es gibt p Objekte der Implementierungsklasse DF_PT und a Objekte der Implementierungsklasse DF_AD, in die VAS-Applika-

tionen der Applikationsklassen Punktespeicher und Ticket bzw. Ausweis und Datenspeicher geladen werden können.

Speziell für die Applikationsklasse Gutschein, die öffentliche Lesezugriffe für alle VAS-Teilnehmer zum gemeinsamen Datenaustausch enthalten soll, wird die Implementierungsklasse Transferfeld verwendet. Schreibzugriffe sind ausschließlich indirekt durch die Anwendung des TRANSFER Befehls möglich, der eine Signatur mit einem korrekten K_{DEC} verlangt. Diese Implementierungsklasse besteht aus genau einem Eintrag im Elementary File EF_TRANSFER, das zu den globalen Daten des VAS-Container gehört. Die Objekte dieser Klasse sind Records in dieser Datei. Wir nennen diese Implementierungsklasse auch "EF_TRANSFER".

Es gibt die in **Fig.** 7 dargestellten Implementierungsklassen innerhalb des VAS-Container. Diese Implementierungsklassen bilden das Speichermodell des VAS-Container.

Die Bereitstellung dem Speichermodells kann auf zwei Arten erfolgen. Die Wahl hängt von der Kartenplattform ab:

- Feste Partitionierung des Speicherbereichs für den VAS-Container:
  Für die Implementierungsklassen DF_PT und DF_AD wird vom Issuer jeweils ihre maximale Anzahl p bzw. a von Objekten festgelegt und diese vom Issuer auf die Karte mit CREATE FILE Befehlen geladen. Die Objekte notieren wir mit DF_PT_i bzw. DF_AD_i. VAS-Applikationen können zu einem späteren Zeitpunkt in freie, also nicht von anderen VAS-Applikationen belegte Objekte geladen werden. Zum Laden bzw. Löschen von VAS-Applikationen sind dann nur UPDATE RECORD Befehle notwendig.
  Der Issuer legt also die Anzahl von möglichen Objekten der beiden Implementierungsklassen nach seinen eigenen Bedürfnissen fest; dies kann eventuell nicht mit dem tatsächlichen VAS-Nutzungsverhalten des Karteninhabers übereinstimmen. Die Aufteilung wird über den gesamten Lebenszyklus der Karte beibehalten. Eine VAS-Applikation wird in ein Objekt geladen, das von einer geeigneten Implementierungsklasse ist. So kann z. B. eine VAS-Applikation, die einen Ausweis darstellt, auch in einem Objekt der Implementierungsklasse DF_PT untergebracht werden, wenn keine Objekte der Implementierungsklasse DF_AD (mehr) zur Verfügung stehen. Die Zuweisung einer VAS-Applikation zu einem Objekt erfolgt durch den Eintrag eines Tripels (PIX der VAS-Applikation, FID des belegten Objektes, Klartextname der Applikation) in die globale Datei EF_DIR des VAS-Container. Das Entfernen der Applikation entspricht dem Entfernen dieses Tripels aus EF_DIR und dem zerstörenden Lesen (durch UPDATE RECORD Befehle) des von der Applikation belegten Speichers.
  Diese Variante findet Verwendung bei Kartenplattformen, die keine dynamische Erzeugung bzw. Löschung von DF/EF Strukturen unterstützen.
- Dynamisches Anlegen von Objekten der Implementierungsklasse:
  Für jede zu ladende VAS-Applikation werden mit CREATE FILE Befehlen die für die zugrundeliegende Implementierungsklasse benötigten Dateien angelegt. Beim Löschen der VAS-Applikation wird die von ihr belegte DF/EF vollständig aus der Karte entfernt und der Speicherplatz freigegeben. Die maximale Anzahl von Objekten von Implementierungsklassen ist hier nicht explizit vom Issuer festzulegen und hängt nur vom zur Verfügung stehenden Speicherplatz der Karte ab.
  Diese Variante erfordert eine dynamische Speicherverwaltung durch das Kartenbetriebssystem.

Im vorliegenden Ausführungsbeispiel gehen wir von der ersten Variante aus, da die zweite erhöhte Anforderungen an die zur Verfügung stehende Kartenplattform stellt. Steht eine dynamische Speicherverwaltung jedoch zur Verfügung und ist sichergestellt, daß sich durch das dynamische Anlegen von Objekten mehr Speicher sparen läßt als die Verwaltung kostet, dann kann das bestehende Konzept übernommen werden und bietet dem Karteninhaber mehr Flexibilität.

Im folgenden werden die Datenstrukturen und Kommandos vorgestellt, mit denen sich der VAS-Container und die Funktionalität der VAS-Kundenkarte gegenüber anderen Systemkomponenten realisieren lassen. Außerdem werden für typische Geschäftsfälle VAS-Operationen festgelegt, die die jeweilige Interaktion zwischen VAS-Container und Terminal beschreiben.

Für die Implementierung wird von folgenden Voraussetzungen ausgegangen:

- An einem Händlerterminal stellt jede Karte, unabhängig von der Plattform, die gleiche Daten- und Kommandoschnittstelle zur Verfügung. Die erforderlichen Kommandos sind daher in ihrer Kodierung eindeutig beschrieben.
- Serviceterminals sind fähig, die Kartenplattform zu erkennen. Die Funktionalität kann daher durch spezifische Kommandos der Plattform erreicht werden. Diese Vorgänge sind daher teilweise informell beschrieben. Wie diese Funktion bereitgestellt wird ist dem Kartenhersteller freigestellt.

In **Fig.** 7 werden hierzu schematisch verschiedene Implementierungsklassen des VAS-Containers dargestellt. **Fig.** 8 zeigt schematisch die Dateistruktur des VAS-Containers, **Fig.** 9 zeigt schematisch die Dateistruktur der Implementierungsklasse DF_PT, **Fig.** 10 die Dateistruktur der Implementierungsklasse DF_AD.

Der Zugriff auf die Dateien des VAS-Container wird explizit durch folgende Access Conditions (AC) eingeschränkt:

Tabelle 4

Zugriffsbedingungen

Datei	ADMIN	Lesezugriff	Schreibzugriff
DF_VAS	K_{SO}	NEV	NEV
EF_ID	K_{SO}	ALW	K_{SO}
EF_DIR	K_{SO}	ALW	K_{SO}
globale KEYs	K_{SO}	NEV	K_{SO}
PIN	K_{SO}	NEV	K_{SO}
EF_VERSION	K_{SO}	ALW	K_{SO}
EF_SEQ	K_{SO}	ALW	K_{SO}
EF_TRANSFER	K_{SO}	ALW	K_{SO}
DF_X ($X=PT_1,...,PT_p,AD_1,...,AD_a$)	K_{SO}	NEV	NEV
EF_KEY	K_{SO}	NEV	K_{SO}
EF_INFO	K_{SO}	PIN oder K_{RVASP} oder K_{SO}	K_{LVASP}
EF_INTERNAL	K_{SO}	K_{LVASP}	K_{LVASP}
EF_VALUE	K_{SO}	PIN oder K_{RVASP} oder K_{SO}	K_{LVASP}

Dabei haben die AC folgende Bedeutung:

- ALW (Always) = Der Zugriff des Kommandos auf die Datei ist immer erlaubt.
- NEV (Never) = Der Zugriff des Kommandos auf die Datei ist nie erlaubt.
- K_{SO} = Vor Zugriff muß externe Authentifizierung des System Operator mit dem Schlüssel K_{SO} erfolgen.
- K_{LVASP} = Vor Zugriff muß externe Authentifizierung des VAS-Provider mit dem Schlüssel K_{LVASP} erfolgen.
- K_{RVASP} = Vor Zugriff muß externe Authentifizierung eines vom VAS-Provider autorisierten Terminals mit dem Schlüssel K_{RVASP} erfolgen.
- PIN = Vor Zugriff muß die korrekte PIN durch den Karteninhaber eingegeben werden und im Klartext mit dem Kommando VERIFY an die Karte geschickt werden.
- PIN oder K_{RVASP} oder K_{SO} = Vor Zugriff muß entweder die korrekte PIN durch den Karteninhaber eingegeben werden, oder es erfolgt eine externe Authentifizierung durch den VAS-Provider mit K_{RVASP} oder durch den System Operator mit K_{SO}.

Dabei ist anzumerken, daß die Oder-Verknüpfung von Zugriffsrechten, wie oben beschrieben, in Kartenbetriebssystemen in der Regel nicht vorgesehen ist. Eventuell bedeutet dies besonderen Implementierungsaufwand. (Alternative: spezielles READ Kommando mit implizit festgelegtem Sicherheitsattribut).

Datenfelder innerhalb der Records von Elementary Files werden nach folgenden Formaten unterschieden: ASCII, Binär, BCD, Datum, Formatstring.

Datenelemente vom Typ "Formatstring" enthalten VAS-Daten in gepackter Darstellung, die dem Karteninhaber am Terminal angezeigt werden können.

Zur optimalen Speicherausnutzung werden Klartext und binäre Daten gemischt und über Formatiermakros darstellbar gemacht.

Sämtliche Elementary Files (EF) des VAS-Container sind nach ISO 7816-4 definiert als lineare formatierte EF Dateien mit Records konstanter Länge (linear fixed record files).

Das Elementary File EF_ID des DF_VAS besteht aus einem Record, der die VAS-Container ID enthält.

Das Elementary File EF_DIR des DF_VAS besteht aus nr_{DIR} Records. Für jede in den VAS-Container geladene VAS-Applikation wird in einem Record des EF_DIR ihr Proprietary Identifier Extension (PIX) und ihr physikalischer Speicherplatz (FID des DF_X, in den die Applikation geladen wurde) festgehalten. Die PIX identifiziert z. B. als Kennziffer die Applikation und den Service-Provider, dem die Applikation zugeordnet ist.

Die Einträge im EF_DIR werden am Service Terminal des System Operators dynamisch angelegt. Records ohne Eintrag werden mit einem leeren TLV Objekt '61' angelegt.

Beim Laden einer VAS-Applikation wird zunächst ein freier Speicherbereich der für sie passenden Implementierungsklasse gesucht, bestimmte VAS-Daten dort eingetragen und schließlich im EF_DIR ein neuer Record erzeugt.

Beim Löschen einer Applikation muß in EF_DIR entsprechend ein leeres TLV Objekt geschrieben werden.

Das Elementary File EF_VERSION des DF_VAS besteht aus einem Record, der eine Versionsnummer des VAS-Containers enthält.

Die Versionsnummer kann von Terminals dazu benutzt werden, verschiedene VAS-Container-Varianten und/oder verschiedene Software-Versionen zu unterscheiden.

Das Elementary File EF_SEQ des DF_VAS besteht aus einem Record, der die Nummer des nächsten Transferfeldeintrags enthält.

Die Sequenznummer wird vom Ergänzungskommando TRANSFER ausgelesen. Dieses Kommando erzeugt daraufhin im Transferfeld EF_TRANSFER einen Record, in den u. a. die Sequenznummer aus EF_SEQ übertragen wird. Zusammen mit dem Kommando TAKE, das sicherstellt, daß jeder Record aus dem Transferfeld nur einmal entnommen wird und diese Entnahme per Signatur über die Sequenznummer dokumentiert, kann die einmalige Entnahme von (Original-) Gutscheinen bzw. Quittungen garantiert werden.

Im folgenden soll nun genauer auf den Transferspeicher eingegangen werden.

Der Transferspeicher wird innerhalb des VAS-Containers angelegt und umfaßt $nr_{EF\_TRANSFER}$ Records. Einträge in den Records dieses Files enthalten Transferdaten, die vom TRANSFER Kommando erzeugt werden. Über den Transferspeicher wird sowohl der Datenaustausch zwischen VAS-Applikationen als auch die Speicherung von VAS-Daten der Klasse Gutschein abgewickelt.

Der Transferspeicher kann ohne Beschränkung gelesen werden, der schreibende Zugriff ist jedoch nur durch die VAS-spezifischen Kommandos TRANSFER und TAKE möglich.

Die Belegung der Datenfelder durch den TRANSFER Befehl geschieht durch einen "Last-recently-used" Algorithmus in der Karte. Der älteste Record in der Datei kann durch Suchen des kleinsten Wertes in den ersten 2 Bytes des Records ermittelt werden.

Datenfelder können mit dem Befehl "TAKE" im Transferspeicher als entnommen und/oder entfernt markiert werden. In jedem Fall erfolgt das Löschen von Daten im Transferspeicher durch Überschreiben mit neuen Daten.

Jeder Record im Transferspeicher enthält beispielsweise ein Verfallsdatum, die Terminal-ID, die PIX, die Sequenznummer und eventuell weitere Daten.

Jedes Elementary File EF_INFO innerhalb von Verzeichnissen DF_X (mit $X = PT_1, \ldots, PT_p, AD_1, \ldots, AD_a$) besteht aus einem Record, der das Verfallsdatum sowie allgemeine VAS-Daten einer VAS-Applikation enthält. Beispielsweise kann die Art eines Tickets (Einzel- oder Streifenfahrschein) oder der Einsteigeort hier angegeben werden. EF_INFO muß jedoch zumindest einen Klartextnamen der Applikation enthalten, der für die Operation Ansehen VAS-Applikationen auslesbar ist. Sensible Daten muß der VAS-Provider durch geeignete externe Verschlüsselungsalgorithmen schützen.

Dieses EF ist lesbar, wenn eine externe Authentifizierung mit K_{SO} oder K_{RVASP} erfolgt oder der Karteninhaber im Falle eines aktivierten PIN-Schutzes eine korrekte PIN eingibt.

Jedes Elementary File EF_INTERNAL innerhalb von Verzeichnissen DF_X (mit $X = PT_1, \ldots, PT_p, AD_1, \ldots, AD_a$) besteht aus einem Record, der interne VAS-Daten des VAS-Provider über seine geladene VAS-Applikation enthalten kann. Weder Karteninhaber noch ein anderer VAS-Provider können diese internen Daten lesen.

Jedes Elementary File EF_VALUE innerhalb von Verzeichnissen DF_X (mit $X = PT_1, \ldots, PT_p, AD_1, \ldots, AD_a$) besteht aus einem Record, der ein ganzzahliges Wertfeld der VAS-Daten enthalten kann. Dieses EF ist lesbar, wenn eine externe Authentifizierung mit K_{SO} oder K_{RVASP} erfolgt oder der Karteninhaber im Falle eines aktivierten PIN-Schutzes eine korrekte PIN bzw. ein korrektes Paßwort eingibt.

Die folgenden Schlüsselfelder werden vom VAS-Container oder von der VAS-Applikation benutzt. Im folgenden wird von einer reinen DES-Verschlüsselung ausgegangen. D.h. sämtliche Schlüssel des VAS-Containers sind DES-Schlüssel und sind in 8 byte codiert (einschließlich Paritätsbits).

Innerhalb des VAS-Containers und innerhalb der VAS-Applikationen können folgende Schlüssel durch ihren KID (Key Identifier) referenziert werden:

Tabelle 5

Schlüssel VAS-Container

KEY	KID
K_{SO}	'00'
K_{AUT}	'01'
K_{SIG_VASC}	'02'
KGK_{DEC}	'03'

Jeder Schlüssel ist mit einem Fehlbedienungszähler verknüpft. Dieser registriert fehlgeschlagene Authentifizierungsversuche mit diesem Schlüssel und blockiert eine weitere Nutzung, wenn ein Limit für diesen Zähler eingetragen und erreicht ist.

Innerhalb der VAS-Applikation können folgende Schlüssel durch ihren KID referenziert werden.

Tabelle 6

Schlüssel VAS-Applikation

KEY	KID
K_{LVASP}	'04'
K_{RVASP}	'05'

Jeder Schlüssel ist mit einem Fehlbedienungszähler verknüpft. Dieser registriert fehlgeschlagene Authentifizierungsversuche mit diesem Schlüssel und blockiert seine weitere Nutzung, wenn ein Limit für diesen Zähler eingetragen und erreicht ist.

Der VAS-Container enthält eine persönliche Identifikationsnummer oder Paßwort (PIN). Dies wird vom VERIFY Kommando zur Identifikation des Karteninhabers benutzt. Die PIN ist verknüpft mit einem Fehlbedienungszähler, der jede Falscheingabe registriert und bei Erreichen eines Limits den PIN-Vergleich sperrt. Diese Sperre kann durch den System Operator mit geeigneten Administrationskommandos zurückgesetzt werden. Der Fehlbedienungszähler wird bei korrekt eingegebener PIN zurückgesetzt.

Es gibt folgende Parameter für den VAS-Container, die vom Kartenherausgeber gemäß dem zur Verfügung stehenden Platz auf einer Kartenplattform und seinen individuellen Wünschen gewählt werden können:

- p Maximale Zahl von Objekten der Implementierungsklasse DF_PT = maximale Zahl von VAS-Applikationen der Applikationsklassen Punktespeicher und Ticket, die gleichzeitig in einen VAS-Container geladen werden können;
- a Maximale Zahl von Objekten der Implementierungsklasse DF_AD = maximale Zahl von VAS-Applikationen der Applikationsklassen Ausweis und Datenspeicher, die gleichzeitig in einen VAS-Container geladen werden können;
- nr_{DIR} Maximale Gesamtzahl von Objekten: $nr_{DIR} = p + a$. Die Anzahl von Records in EF_DIR ist nr_{DIR};
- $nr_{EF\_TRANSFER}$ Anzahl von Records des EF_TRANSFER.

Der Speicherbedarf für die Größen der beschriebenen Elementary Files ist:

		Byte
Globale Daten des VAS-Containers:	EF_ID	4
	EF_DIR	9 * (p+a)
	EF_VERSION	1
	EF_SEQ	2
	Globale Keys, Pin	64+2
Transferfeld:	EF_TRANSFER	48 * $nr_{EF\_TRANSFER}$
Proprietäre Daten der VAS-Provider:	EF_KEY	(p+a) * 32
	EF_INFO	(p+a) * 62
	EF_INTERNAL	p * 10
	EF_VALUE	p * 3

Wählen wir z. B. für die Parameter p, a und $nr_{EF\_TRANSFER}$ folgende Werte, dann ergibt sich folgender Mindestspeicherbedarf für den VAS-Container (ohne Speicherbedarf für Ergänzungskommandos):

Parameter	Speicherbedarf
p = 8, a = 3, $nr_{EF\_TRANSFER}$ = 15	2030 Byte
p = 10, a = 5, $nr_{EF\_TRANSFER}$ = 20	2758 Byte.

Der Maximalspeicherbedarf ist wahrscheinlich um ca. 10% höher als der Mindestspeicherbedarf.

Im folgenden wird nun genauer auf die Kommandos der erfindungsgemäßen Chipkarte eingegangen.

Das READ RECORD Kommando wird zum Auslesen von Daten aus linearen Elementary Files benutzt. Die Karte liefert in der Rückantwort den Inhalt des Records. Das EF wird durch den Short File Identifier (SFI) referenziert.

Der Status Code '9000' signalisiert eine erfolgreiche Durchführung des Kommandos; jeder andere Code wird als Fehler bewertet.

Der UPDATE RECORD Befehl wird benutzt, um Daten in die Records eines linearen EF einzubringen. Die Kommandonachricht enthält die Referenz auf das EF, den Record und die Daten.

Die Antwort der Karte enthält den Statuscode. Der Statuscode '9000' signalisiert den erfolgreichen Abschluß der Kommandos. Andere Statuscodes zeigen den Fehlerfall an.

Das GET CHALLENGE Kommando fordert von der Karte eine Zufallszahl an. Die Zufallszahl wird im Zusammenhang mit der dynamischen Authentifizierung im EXTERNAL AUTHENTICATE verwendet.

Die Antwortnachricht der Karte enthält eine 8 bytes lange Zufallszahl und den Statuscode. Der Statuscode '9000' zeigt die erfolgreiche Ausführung des Kommandos an. Andere Statuscodes zeigen den Fehlerfall an.

Das Kommando EXTERNAL AUTHENTICATE erlaubt die Authentifizierung eines Terminals gegenüber der Karte. EXTERNAL AUTHENTICATE wird im Rahmen von VAS-Applikationen zur Authentifizierung des System Operators und des VAS-Providers benutzt. EXTERNAL AUTHENTICATE überträgt ein Kryptogramm in die Karte, das zuvor vom Terminal durch Verschlüsseln einer Zufallszahl erzeugt werden muß. Die Karte vergleicht das Kryptogramm mit einem Referenzwert, den sie selbst nach dem gleichen Verfahren berechnet hat. Sind beide Werte gleich, so vermerkt die Karte intern, daß die Zugriffsbedingung Authentifizierung für diesen Schlüssel erfolgt ist. Sollte der Vergleich negativ ausfallen, so gibt die Karte den Status "Nicht autorisiert" zurück und dekrementiert einen internen Fehlbedienungszähler. Erreicht dieser Zähler den Wert 0, so wird jede weitere Ausführung des EXTERNAL AUTHENTICATE mit dem Status "Authentifizierung gesperrt" abgewiesen.

Die Antwortnachricht der Karte enthält den Statuscode. Die erfolgreiche Durchführung des Kommandos und die Authentifizierung der Terminals gegenüber der Karte wird durch den Statuscode '9000' angezeigt. Andere Statuscodes zeigen den Fehlerfall an.

Das Kommando INTERNAL AUTHENTICATE wird benutzt, um die Echtheit des VAS-Containers durch ein Terminal prüfen zu können. Die Karte berechnet dazu ein Kryptogramm über Referenzdaten vom Terminal. Das Terminal bildet seinerseits das Kryptogramm und vergleicht es mit dem Wert von der Karte. Bei Gleichheit kann das Terminal die Echtheit des VAS-Containers annehmen.

Die Antwort der Karte enthält das Kryptogramm und den Statuscode für die Ausführung des Kommandos. Die erfolgreiche Durchführung des Kommandos wird durch den Statuscode '9000' angezeugt. Andere Statuscodes zeigen den Fehlerfall an.

Das VERIFY Kommando wird zur Verifikation der Karteninhaber PIN benutzt. Das Kommando überträgt die PIN Daten unverschlüsselt an die Karte, wo ein Vergleich mit einem gespeicherten Referenzwert durchgeführt wird. Sind eingegebener und gespeicherter Wert identisch, so wird die Zugriffsbedingung "PIN" als erfüllt betrachtet.

Die Antwortnachricht der Karte enthält den Statuscode. Der Statuscode '9000' zeigt die erfolgreiche Durchführung des Kommandos an. Andere Statuscodes zeigen den Fehlerfall an.

Das TRANSFER Kommando erzeugt einen Eintrag im Transferspeicher. Drei Arbeitsmodi sind für das Kommando definiert:

1. Erzeugen eines Eintrags im Transferfeld durch Verringern von Werten im EF_VALUE Feld der Applikation der Klasse Ticket oder Punktespeicher.
2. Erzeugen eines Eintrags im Transferfeld durch Erstellen einer Quittung in Applikationen der Klasse Ausweis.
3. Erzeugen eines Eintrags im Transferfeld durch Erstellen eines Gutscheins in Applikationen der Klasse Gutschein.

Der Arbeitsmodus wird durch die Karte automatisch ausgewählt: Wird der Befehl innerhalb eines selektierten Applikations-DF ausgeführt, so wird zunächst das Vorhandensein eines EF_VALUE überprüft. Bei vorhandenem EF_VALUE führt die Karte den Befehl im Modus 1, ansonsten im Modus 2 durch. Ist innerhalb des VAS-Containers kein Applikations-DF ausgewählt, wird Modus 3 benutzt.

Das Terminal versorgt die Karte beim Aufruf des TRANSFER Kommandos mit den folgenden Daten:

- Aktuelles Datum
- Verfallsdatum für den Eintrag im Transferfeld
- Identifikation für das Terminal, welches diesen Eintrag erzeugt
- Nutzdaten für das Transferfeld
- PIX der VAS-Applikation (nur Modus 3)
- Anzahl abzuziehender Einheiten (nur Modus 1)
- MAC über die o.g. Daten, die Sequenznummer und die VAS-Container Nummer.

Die Karte führt beim Aufruf des TRANSFER Kommandos die folgende Sequenz aus:

1. Suchen nach einem freien Eintrag im Transferspeicher. (Die folgende Aufzählung gibt absteigend die Priorität wieder, in welcher vorhandene Eintragungen überschrieben werden sollen: "Eintrag als entfernt markiert", "Eintrag als entnommen markiert" und "Verfallsdatum erreicht", "Verfallsdatum erreicht").
2. Anhängen der PIX an die Daten vom Terminal in den Modi 1 und 2.
3. Anhängen der Sequenznummer an die Daten aus Schritt 2.
4. Anhängen der VAS-Container ID an die Daten aus Schritt 3.
5. Ableiten des $KGK_{DEC,PIX}$ durch Verschlüsseln der PIX unter dem KGK_{DEC}.
6. Ableiten des K_{DEC} durch Verschlüsseln der Terminal ID unter dem $KGK_{DEC,PIX}$.
7. Erzeugung des MAC über die Daten aus Schritt 4.
8. Vergleich des MAC aus Schritt 7 mit dem MAC vom Terminal. Sind die Werte unterschiedlich, bricht die Karte die Funktion ab und verringert den Fehlbedienungszähler für den KGK_{DEC}.

9. Für Modus 1: Prüfung des Wertfeldes EF_VALUE im Verzeichnis, in das die Applikation geladen wurde. Sind nicht genügend Einheiten in diesem Feld vorhanden, bricht die Karte die Funktion an dieser Stelle ab. Andernfalls wird das Wertfeld in der Applikation um den Betrag vermindert.
10. Zusammenstellen der Kommandonachricht.
11. Inkrementieren des Inhalts von EF_SEQ um 1.

Die Kommandonachricht enthält beispielsweise Felder mit einem Verfallsdatum, der Terminal ID, den Transaktionsdaten, ein Feld für den Betriebsmodus (enthält z. B. im Modus 3 die PIX), sowie ein Kryptogramm.

Das Kryptogramm wird unter Verwendung des Schlüssels K_{DEC} berechnet, wobei die Daten, über die der MAC gebildet wird, beispielsweise die Transaktionsdaten und die Terminal-ID umfassen.

Die Antwortnachricht des TRANSFER Kommandos umfaßt im Erfolgsfall einen 8 Byte langes Datenfeld und den 2 Byte langen Statuscode '9000'. Antwortnachrichten mit einem von '9000' verschiedenen Statuscode werden als Fehlercode interpretiert. Das Datenfeld der Antwortnachricht enthält im fehlerfreien Fall (d. h. insbesondere war das Kryptogramm der Kommandonachricht korrekt) das mit K_{DEC} verschlüsselte Kryptogramm der Kommandonachricht. Dadurch wird implizit (anstelle einer internen Authentifizierung mit K_{AUT}) ein Echtheitsnachweis durch den VAS-Container erbracht.

Das Kommando TAKE dient zur Entnahme von Objekten aus der Implementierungsklasse EF_TRANSFER. Technisch bedeutet die Ausführung des Kommandos TAKE das Auslesen eines anzugebenden Records aus der Datei EF_TRANSFER, wobei der Record in der Datei so lange verbleibt, bis der Speicherplatz für neue Einträge benötigt wird, der Datensatz wird als entnommen markiert. Technisch gesehen kann jeder das Kommando TAKE nutzen, um einen Datensatz zu entnehmen, nach den R&R sollte dies jedoch nur ein VAS-Provider tun, für den der Datensatz bestimmt ist.

Für den Entnahmevorgang kann folgendes angenommen werden. Der VAS-Provider, der einen Gutschein oder eine Quittung entnehmen will, durchsucht den Transferspeicher nach einem passenden Datensatz (z. B. durch das Kommando SEEK oder durch explizites Lesen jedes Records). Der Datensatz wird in jedem Fall gelesen und eventuell inhaltlich geprüft.

Die Ausgabe des Datensatzes in der Antwort ist daher nicht notwendig. Es kann weiterhin angenommen werden, daß die Nummer des Records bekannt ist.

Das Kommando TAKE stellt folgende Modi bereit:

- Entnahme mit gleichzeitigem Vermerk, daß die Daten ungültig geworden sind.
- Entnahme ohne vorstehenden Vermerk. Die Daten bleiben weiterhin gültig.

Die Kommandonachricht enthält Felder mit Terminal-ID, PIX der Applikation und eine vom Terminal generierte Zufallszahl.

Die PIX im Kommando bezeichnet die entnehmende Anwendung. Sie darf verschieden sein von der PIX des Datensatzes, der entnommen wird. Sie dient ausschließlich zur Ableitung von K_{DEC} des entnehmenden Händlerterminals.

Die Antwortnachricht der Karte enthält ein Kryptogramm C_1 mit K_{SIG_VASC} über die Daten des in der Kommandonachricht angegebenen Records des Transferfeldes und die VAS-Container ID, ein Kryptogramm C_2 mit K_{DEC} des entnehmenden Terminals über C_1 und die Zufallszahl aus der Kommandonachricht. Die Antwortnachricht enthält zusätzlich den Statuscode. Der Schlüssel K_{DEC} wird wie vorher beschrieben abgeleitet. Mit dem Kryptogramm vermöge K_{DEC} wird implizit ein Echtheitsnachweis durch den VAS-Container erbracht. Mit dem Kryptogramm C wird ein Echtheitsnachweis und Einmaligkeitsnachweis (da C über Sequenznummer, Entnommen-Bit des Transferfeld-Records und VAS-Container ID gebildet werden) berechnet, den der Systemoperator verifizieren kann.

Der Statuscode '9000' zeigt die erfolgreiche Durchführung des Kommandos an. Andere Statuscodes werden als Fehler interpretiert.

Es ist davon auszugehen, daß der SO eine AID_{VAS} gemäß ISO/IEC 7816-5 beantragt. Genauer: Er beantragt eine 5 Byte lange RID_{VAS} für das VAS-System.

Die AID_{VAS} des Verzeichnisses DF_VAS soll lauten: $AID_{VAS} = RID_{VAS} \cdot PIX_{DF\_VAS}$.

Für jede VAS-Applikation A wird gemäß R◇ eine 3 Bytes lange PIX_A vergeben, um sie innerhalb des VAS-Container eindeutig mit $AID_A = RID_{VAS} \cdot PIX_A$ identifizieren zu können. Nach der Selektion des DF_VAS kann ein Verzeichnis, in dem die VAS-Applikation A enthalten ist, mit SELECT FILE <PIX_A> ausgewählt werden.

Mit dem UPDATE KEY(KID, K) sei ein von der Kartenplattform abhängiges Kommando bezeichnet, mit welchem ein Schlüssel mit Key Identifier ID durch den neuen Wert K ersetzt wird.

Bevor eine VAS-Applikation aus einer der Implementierungsklassen DF_PT oder DF_AD (dies entspricht den Applikationsklassen Punktespeicher, Ticket, Ausweis oder Datenspeicher) durch den Karteninhaber an Händlerterminals genutzt werden kann, muß sie an einem Serviceterminal des zuständigen VAS-Provider in den VAS-Container geladen werden. Prinzipiell ist auch möglich, daß ein Kartenherausgeber im Auftrag eines VAS-Provider und eines SO bereits beim Aufbringen des VAS-Containers eine oder mehrere VAS-Applikationen in den VAS-Container lädt. Dieser Ladevorgang ist ein Spezialfall des hier beschriebenen.

Ablauf des Ladens einer VAS-Applikation:

1. Karteninhaber führt eine VAS-Karte in ein Serviceterminal ein.
2. Das Serviceterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container Nummer).

   Optional wird die Gültigkeit des VAS-Containers geprüft. Dazu verlangt das Serviceterminal eine interne Authentifizierung des VAS-Containers:
   - INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{AUT}>

Das Serviceterminal prüft die Antwort und bricht im Fehlerfall (mit Fehlermeldung) ab.

3. Das Serviceterminal stellt dem Karteninhaber mehrere Optionen zur Auswahl. Eine davon lautet Laden VAS-Applikation. Diese wählt der Karteninhaber aus. Nun werden dem Karteninhaber alle VAS-Applikationen, die am Serviceterminal geladen werden können, angezeigt, und es wird auf eine Auswahl gewartet. Dazu wird die Operation Ansehen VAS-Applikationen aktiviert. Der Karteninhaber wählt eine VAS-Applikation A der Implementierungsklasse DF_PT oder DF_AD aus.

4. Das Serviceterminal untersucht den VAS-Container mit der Operation Selektion einer VAS-Applikation, ob die ausgewählte VAS-Applikation mit PIX_A bereits in der Karte geladen ist. Falls ja, wird eine Fehlermeldung ausgegeben. Falls nicht, wird geprüft, ob noch ein Objekt der für A geeigneten Implementierungsklasse im VAS-Container frei ist. Dies geschieht durch Suchen eines freien Records in EF_DIR (z. B. mit dem Kommando SEEK). Falls nichts frei ist, wird eine Fehlermeldung ausgegeben. Falls ein Record frei ist, enthält dieser eine FID_{DF_X} eines DF_X, in das keine VAS-Applikation geladen ist.

5. Das nächste freie Objekt der für A geeigneten Implementierungsklasse wird belegt für die VAS-Applikation A. Dabei erfragt zunächst das Serviceterminal offline vom VAS-Provider (z. B. durch ein VAS-Provider-SAM) zwei Schlüssel K_{LVASP} und K_{RVASP} und weist sie der neuen VAS-Applikation zu:

- SELECT FILE <FID_{DF_X}>
- GET CHALLENGE
- EXTERNAL AUTHENTICATE <K_{SO}(Zufallszahl), KID von K_{SO}>
- UPDATE KEY <KID von K_{LVASP}, K_{LVASP}>
- UPDATE KEY <KID von K_{RVASP}, K_{RVASP}>
- GET CHALLENGE
- EXTERNAL AUTHENTICATE <K_{LVASP} (Zufallszahl), KID von K_{LVASP}>
- UPDATE RECORD <SFI von EF_INFO des DF_X, Daten>
- UPDATE RECORD <SFI von EF_INTERNAL des DF_X, Daten>
- Optional (Initialisierung): UPDATE RECORD <SFI von EF_VALUE des DF_X, Daten>.

6. Nach dem erfolgreichen Schreiben in die EFs wird vom Serviceterminal die Operation Eintrag VAS-Applikation <PIX_A, FID_{DF_X}> ausgeführt, die das DF_X mit dem PIX_A verbindet und so SELECT FILE mittels des PIX_A (nach der vorherigen Selektion mit SELECT FILE AID_{VAS}) ermöglicht.

Der mit dem Transferspeicher zur Verfügung gestellte Mechanismus läßt sich z. B. von VAS-Providern nutzen, die Intra- oder Interservices ohne proprietäre DF-Strukturen abwickeln möchten. Ein Karteninhaber muß dann insbesondere vor der Benutzung einer VAS-Applikation diese nicht erst an einem Serviceterminal laden. Er kann dagegen direkt am Händlerterminal sich einen Gutschein oder eine Quittung ausstellen und dies an einem anderen Terminal erstatten (durch die Operation Entnehmen) lassen oder den Gutschein bzw. die Quittung nur einfach vorweisen (durch Lesen). Das Laden einer VAS-Applikation der Implementierungsklasse EF_TRANSFER wird daher implizit durch Aufbringen von VAS-Daten realisiert bzw. auf diese Operation zurückgeführt. Hierzu wird auf die später folgende Beschreibung des Erwerbens der Implementierungsklasse EF_TRANSFER verwiesen.

Nachfolgend wird der Ablauf der Operation des Eintrags einer VAS-Applikation beschrieben.

Ist eine VAS-Applikation in den VAS-Container geladen, ist einem Terminal der physikalische Ort unbekannt, in welchem DF_X mit $X = PT_1, \ldots, PT_p, AD_1, \ldots, AD_a$ die VAS-Applikation bzw. ob sie überhaupt geladen ist. Aus Sicht des Kartenherstellers sind zwei mögliche Implementierungsweisen möglich, die getrennt geprüft werden sollen; dabei ist insbesondere die Konsistenz zum ZKA-Standard zu beachten.

Der erste Fall: Zugriff auf EF_DIR möglich

Folgende Voraussetzungen sind zu erfüllen:

- Es gebe standardmäßig in der Kartenplattform ein EF_DIR (z. B. unter DF_VAS), in dem AIDs den FIDs der DF_X zugeordnet sind, in denen sich VAS-Applikationen aktuell befinden.
- Dieses EF_DIR ist für jeden lesbar (AC von READ RECORD: ALW).
- Wird eine VAS-Applikation A mit PIX_A vom System Operator in ein freies (unbenutztes) DF_X geladen, d. h. die vorhandenen Elementary Files dieses DF_X beschrieben (kein CREATE FILE !), dann soll durch ein UPDATE RECORD die Datei EF_DIR um den Eintrag PIX_A erweitert werden können, der auf dieses DF_X verweist mit FID_X. Die AC von UPDATE RECORD sollte daher eine externe Authentisierung mit dem Schlüssel K_{SO} erzwingen.
- Wird der Befehl SELECT FILE <PIX_A> nach der vorherigen Selektion von DF_VAS an die Karte übermittelt, muß das DF_X, in das eine VAS-Applikation A geladen ist, direkt selektierbar sein.
- Wenn eine VAS-Applikation A, die in DF_X und den darunterliegenden Elementary Files geladen ist, auf Wunsch des Karteninhabers an einem Service-Terminal gelöscht werden soll, werden die entsprechenden Dateien nicht mit DELETE FILE gelöscht, sondern die eingetragenen Daten lediglich mit Dummy-Werten überschrieben. Danach soll aus EF_DIR der Eintrag PIX_A (genauer das Paar PIX_A und der Verweis auf DF_X) gelöscht werden können (z. B. durch UPDATE RECORD mit vorheriger externer Authentisierung mit dem Schlüssel K_{SO}).
- Da die Anzahl der DF_X fest ist, ist die Anzahl der Records von EF_DIR bekannt.

Ist dieser Ansatz realisierbar, ergibt sich folgende Konsequenz:

- Ein nach Selektion von DF_VAS direkt erfolgendes Selektieren einer VAS-Applikation mit SELECT FILE PIX_A ist möglich.

Der zweite Fall: Zugriff auf EF_DIR nicht möglich

Steht bei der Kartenplattform kein EF_DIR zur Verfügung oder ist kein lesender und schreibender Zugriff auf dieses EF_DIR – wie im obigen Abschnitt dargestellt – möglich, soll es unter DF_VAS eine Datei EF_VASDIR geben, in der vom System Operator durch explizite UPDATE RECORD Befehle (nach vorheriger externer Authentifizierung mit K_{SO}) ein Bezug der PIX_A geladener VAS-Applikationen zu ihrem physikalischem Speicherplatz in einem DF_X hergestellt wird. Das Lesen und Löschen von Records aus EF_VASDIR soll wie vorher beschrieben möglich sein.

Der Ablauf der Operation Eintrag VAS-Applikation läuft wie folgt:
Eine VAS-Applikation A mit PIX_A wurde vorher in ein freies DF_X mit FID_{DF_X} geladen. Die Nummer des Records mit FID_{DF_X} wurde vom Serviceterminal vermerkt.

1. SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
2. GET CHALLENGE
3. EXTERNAL AUTHENTICATE <K_{SO} (Zufallszahl), KID von K_{SO}>
4. UPDATE RECORD <SFI von EF_DIR des DF_VAS, Nummer Record mit FID_{DF_X}, PIX_A, FID_{DF_X}>.

VAS-Applikationen der Implementierungsklassen DF_PT oder DF_AD können nur an Serviceterminals (da unter Hoheit des System Operator) auf Wunsch des Karteninhabers gelöscht werden, VAS-Applikationen der Implementierungsklasse EF_TRANSFER können überall und von jedem gelöscht werden. Beim Löschen zu unterscheiden sind VAS-Applikationen der Implementierungsklassen DF_PT und DF_AD bzw. der Implementierungsklasse EF_TRANSFER.

Ablauf des Löschens einer solchen VAS-Applikation für die Implementierungsklasse DF_PT oder DF_AD:

1. Der Karteninhaber führt eine VAS-Karte in ein Serviceterminal ein.
2. Das Serviceterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar).
   - READ RECORD <EF_ID> (Auslesen der VAS-Container ID).
3. Das Serviceterminal stellt dem Karteninhaber mehrere Optionen zur Auswahl.
   Eine davon lautet Löschen VAS-Applikation. Diese wählt der Karteninhaber aus. Nun werden dem Karteninhaber alle VAS-Applikationen aller Implementierungsklassen, die im VAS-Container geladen sind, angezeigt, und es wird auf eine Auswahl gewartet. Dazu wird die Operation Ansehen VAS-Applikationen aktiviert. Der Karteninhaber wählt eine VAS-Applikation A der Implementierungsklasse DF_PT oder DF_AD mit AIDA aus. Das Objekt, in dem die VAS-Applikation geladen ist, heiße DF_A.
4. Nach der Selektion des DF_A authentisiert sich das Serviceterminal:
   - SELECT FILE <PIX_A>
   - GET CHALLENGE
   - EXTERNAL AUTHENTICATE <K_{SO} Zufallszahl), KID von K_{SO}>.
5. Nun wird der Inhalt der Dateien aus DF_A gelöscht (für EF_KEY, EF_INTERNAL und EF_VALUE sind der K_{LVASP} notwendig, deshalb wird dieser zuerst vom System Operator gelöscht):
   - UPATE KEY <KID von K_{LVASP}, "00. . .00">
   - UPDATE KEY <KID von K_{RVASP}, "00. . .00">
     GET CHALLENGE
   - EXTERNAL AUTHENTICATE <K_{LVASP} (Zufallszahl), KID von K_{LVASP}>
   - UPDATE RECORD <SFI von EF_INFO des DF_A, "00. . .00">
   - UPDATE RECORD <SFI von EF_INTERNAL des DF_A, "00. . .00">
   - UPDATE RECORD <SFI von EF_VALUE des DF_A, "00. . .00">.
6. Nach einem erfolgreichen Schreiben in die EFs wird abschließend vom Serviceterminal der Eintrag der VAS-Applikation aus EF_DIR gelöscht:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - UPDATE RECORD <SFI von EF_DIR des DF_VAS, Nummer Record mit FID_{DF_A} "00. . .00", FID_{DF_A}>.

Die Verbindung von PIX_A zum DF_A wird dadurch aufgelöst, so daß SELECT FILE mit dem PIX_A nicht mehr möglich ist.

Nun für die Implementierungsklasse EF_TRANSFER:
Eine VAS-Applikation der Implementierungsklasse EF_TRANSFER soll nach R&R nur auf expliziten Wunsch eines Karteninhabers an einem Händlerterminal oder Serviceterminal gelöscht werden (auch wenn dies technisch durch beide möglich wäre). Das Löschen eines Objektes aus EF_TRANSFER wird insbesondere dann notwendig, wenn der Transferspeicher keinen Platz mehr enthält für die Speicherung eines neuen Objektes (z. B. einen Gutschein oder eine Quittung). Das Löschen geschieht immer indirekt über das Ergänzungskommando TAKE. Dieses Kommando markiert nur ein Objekt als entfernt. Damit ist es zum Überschreiben durch ein nachfolgendes Ergänzungskommando TRANSFER freigegeben.

Ablauf des Löschens dieser VAS-Applikation:

1. Der Karteninhaber führt eine VAS-Karte in ein Terminal ein, welches den Inhalt von EF_TRANSFER darstellen kann (Spezialfall von Operation Ansehen VAS-Applikationen).
2. Das Terminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)

– READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID).

3. Das Terminal stellt dem Karteninhaber mehrere Optionen zur Auswahl. Eine davon lautet Löschen VAS-Applikation. Diese wählt der Karteninhaber aus. Nun werden dem Karteninhaber zumindestens die VAS-Applikationen der Implementierungsklasse EF_TRANSFER angezeigt, und es wird auf eine Auswahl gewartet. Dies wird durch einen Spezialfall der Operation Ansehen VAS-Applikationen realisiert. Der Karteninhaber wählt ein Objekt der Implementierungsklasse mit einem Gutschein bzw. einer Quittung aus. Dieses Objekt besitze die Recordnummer A. Der Karteninhaber bestätigt die Auswahl.

4. Das Terminal markiert den Record mit Recordnummer A als entfernt, indem es A, eine von ihm berechnete Zufallszahl, seine PIX und Terminal ID mit dem Kommando TAKE übermittelt:

– TAKE <A, Zufallszahl, PIX, Terminal ID>.

Nun steht der als entfernt markierte Record wieder zur Verfügung zur Aufnahme für einen neuen Gutschein oder eine Quittung.

Nachfolgend der Ablauf der Selektion einer VAS-Applikation:

Ist eine VAS-Applikation A der Implementierungsklassen DF_PT oder DF_AD in den VAS-Container mit der Operation Laden VAS-Applikation geladen worden, kann sie zweistufig von einem Terminal selektiert werden. Zuerst wird der VAS-Container selektiert und danach die VAS-Applikation mit ihrer PIX_A:

1. SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)

Ein Serviceterminal kann die Echtheit des VAS-Containers prüfen. Dazu verlangt ein Serviceterminal eine interne Authentifizierung des VAS-Containers:

– READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID)

– INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{AUT}>.

Das Serviceterminal prüft die Antwort und bricht im Fehlerfall (mit Fehlermeldung) ab.

2. SELECT FILE <PIX_A> (Fehlermeldung, falls VAS-Applikation A nicht in den VAS-Container geladen ist).

Ein Händlerterminal (welches nicht über den KGK_{AUT} verfügt) kann indirekt die Echtheit des VAS-Containers durch die Echtheitsprüfung der VAS-Applikation A feststellen. Denn eine VAS-Applikation kann nur an einem Serviceterminal geladen worden sein, die beim Ladevorgang die Echtheit des VAS-Containers geprüft hat. Die Echtheitsprüfung der VAS-Applikation A kann auf den Test des Händlerterminals zurückgeführt werden, ob der VAS-Container den Schlüssel K_{LVASP} oder den K_{RVASP} für die VAS-Applikation A enthält. Dies kann das Händlerterminal z. B. kontrollieren durch:

– INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{RVASP} oder K_{LVASP}>.

Um zu testen, ob eine VAS-Applikation A im VAS-Container geladen ist, kann sie probeweise selektiert werden; aufgrund der Fehlermeldung der Antwortnachricht von SELECT FILE kann daraus geschlossen werden, daß sie nicht vorhanden ist.

Eine Selektion einer VAS-Applikation der Implementierungsklasse EF_TRANSFER ergibt sich implizit durch die Operation Ansehen VAS-Applikationen und das Speichern der Daten im Terminal. Da das Terminal die Recordnummer von jedem angezeigten Objekt aus EF_TRANSFER kennt, kann per Recordnummer ein beliebiges Objekt zur Weiterverarbeitung durch Bezugnahme auf die Recordnummer selektiert werden.

Nachfolgend der Ablauf des Ansehens einer VAS-Applikation:

Die Operation Ansehen von VAS-Applikationen listet an einem Serviceterminal sämtliche VAS-Applikationen der Implementierungsklassen DF_PT, DF_AD und EF_TRANSFER auf. An einem Händlerterminal können nur die VAS-Applikationen der Implementierungsklasse EF_TRANSFER angezeigt werden, da für diese kein Zugriffsschutz existiert, und optional Applikationen der Implementierungsklasse DF_PT bzw. DF_AD, für die das Händlerterminal Leserechte besitzt (Besitz von K_{RVASP}

Ablauf einer solchen VAS-Applikation:

1. Der Karteninhaber führt eine VAS-Karte (Chipkarte mit gültigem VAS-Container) in das Terminal ein.

2. Das Serviceterminal prüft, ob ein VAS-Container vorhanden ist:

– SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)

– READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID).

Optional wird die Gültigkeit des VAS-Containers geprüft. Dazu verlangt das Serviceterminal eine interne Authentifizierung des VAS-Containers:

– INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{AUT}>.

Das Serviceterminal prüft die Antwort und bricht im Fehlerfall (mit Fehlermeldung) ab.

3. Das Serviceterminal stellt dem Karteninhaber mehrere Optionen zur Auswahl. Eine davon lautet Ansehen VAS-Applikationen. Diese wählt der Karteninhaber aus.

4. Das Serviceterminal authentisiert sich:

– GET CHALLENGE

– EXTERNAL AUTHENTICATE <K_{SO} (Zufallszahl), KID von K_{SO} >.

5. Für VAS-Applikationen der Implementierungsklassen DF_PT und DF_AD werden über den Inhalt von EF_DIR gesteuert nacheinander die einzelnen VAS-Applikationen selektiert und nach einer externen Authentifizierung mit dem Schlüssel K_{SO} der Inhalt der einzelnen EF_INFO (oder ein Teil davon nach R&R) sowie EF_VALUE angezeigt:

– Für i = 0,. . ., nr_{DIR} – 1

- READ RECORD <SFI von EF_DIR, i>.
In der Antwortnachricht steht PIX_A, die "00. . .00" ist, wenn in das entsprechende DF keine VAS-Applikation geladen ist. Alternativ zum READ RECORD aus EF_DIR kann eventuell auch ein SEEK Kommando benutzt werden.
- Wenn PIX_A ungleich "00. . .00"

  - SELECT FILE <PIX_A>
  - SELECT FILE <PIX_A>
  - SELECT FILE <PIX_A>
  - SELECT FILE <PIX_A>
  - READ RECORD <SFI von EF_VALUE, 0>
  - SELECT FILE <AID_{VAS}>.

6. Für VAS-Applikationen der Implementierungsklasse EF_TRANSFER wird der Inhalt (oder ein Teil davon nach R&R) des EF_TRANSFER eines jeden Records gelesen:

- SELECT FILE <AID_{VAS}>

Für $i = 0, \ldots, nr_{EF\_TRANSFER} - 1$

  - READ RECORD <SFI von EF_TRANSFER, i> (in der Antwortnachricht steht Inhalt des i-ten Records von EF_TRANSFER)
  - Wenn Inhalt nicht leer ist, wird der Inhalt interpretiert (z. B. entnommen/verfallen) angezeigt.

Das Ansehen der Applikationen der Implementierungsklasse DF_PT bzw. DF_AD, für die das Händlerterminal Leserechte besitzt (Besitz von K_{RVASP}), erfolgt wie beschrieben – mit zwei Abweichungen: Zur externen Authentifizierung wird anstelle des K_{SO}, über den nur der System Operator verfügt, der Schlüssel K_{RVASP} benutzt. Verfügt ein Händlerterminal nicht über den KGK_{AUT} und möchte dennoch die Echtheit der VAS-Applikationen prüfen, kann das Händlerterminal anstelle der internen Authentifizierung die Authentifizierung (zumindest einer) VAS-Applikation verlangen. Dies geschieht, wie aufgeführt, durch Kenntnis der Karte des entsprechenden K_{RVASP} oder K_{LVASP} Schlüssels.

Für VAS-Applikationen der Implementierungsklasse EF_TRANSFER wird der Inhalt (oder ein Teil davon nach R&R) des EF_TRANSFER jedes Records nacheinander aufgelistet, wie vorher beschrieben.

Die Operation Interpretieren VAS-Applikationen verläuft dazu analog. Das Terminal stellt dazu dem Karteninhaber eine Option Interpretieren VAS-Applikationen zur Auswahl. Zusätzlich zu den Daten, die durch die Operation Ansehen sichtbar werden, können auch Daten aus EF_INFO und EF_VALUE, die einer Interpretation durch den VAS-Provider bedürfen (z. B. extern verschlüsselte Daten) und Daten aus EF_INTERNAL (z. B. Meilen der vergangenen Jahre bei Miles & More), dargestellt werden. Dies ist jedoch nur für die VAS-Applikationen möglich, für die ein VAS-Provider (nach seinen eigenen Vorstellungen) am Terminal geeignete Schlüssel zur Verfügung stellt. Zum Lesen des EF_INTERNAL einer VAS-Applikation ist der Schlüssel K_{LVASP} (externe Authentifizierung) notwendig. Für extern verschlüsselte Daten innerhalb der Elementary Files EF_INFO, EF_INTERN und EF_VALUE sowie des Transferspeichers EF_TRANSFER werden die entsprechenden Schlüssel des VAS-Providers benötigt. Das Terminalprogramm kann anhand der PIX einer selektierten VAS-Applikation leicht entscheiden, für welche Applikationen Schlüssel für die Interpretation der Daten zur Verfügung stehen.

Die Operation Übertragen von VAS-Applikationen bedeutet das Übertragen aller oder ausgewählter VAS-Operationen einer Quell-Karte auf eine Ziel-Karte an einem Serviceterminal. Bedingung ist hierbei, daß im VAS-Container der Zielkarte keine VAS-Applikationen geladen sind und nach der Übertragung auf der Quell-Karte alle VAS-Applikationen gelöscht werden. Beides kann durch sukzessive Anwendung der Operation Löschen einer VAS-Applikation erreicht werden. Außerdem muß die Echtheit der VAS-Karten geprüft werden und die Ziel-Karte über ausreichend Speicherplatz verfügen. Die Operation Übertragung selbst basiert im wesentlichen auf einer Erweiterung der Operation Ansehen von VAS-Applikationen, bei der zusätzlich die Daten aus EF_INTERNAL und die Schlüssel K_{LVASP} und K_{RVASP} gelesen werden, und einer wiederholten Anwendung der Operation Laden einer VAS-Applikation.

Mit den VAS-Daten wird auch die VAS-Container ID mit auf die Ziel-Karte übertragen, damit sich VAS-Applikationen auf der Ziel-Karte genau so verhalten wie auf der Quell-Karte. Der Grund hierfür ist, daß von einem VAS-Provider abgeleitete Schlüssel dich auf die VAS-Container ID stützen, die Schlüssel aber unverändert kopiert werden. Außerdem möchte ein VAS-Provider seine Buchführung im Hintergrundsystem nicht ändern, da er VAS-Karten üblicherweise über die VAS-Container ID identifiziert. Unbedingt zu beachten ist bei der Übertragung der VAS-Container ID, daß diese systemweit eindeutige Nummer auf der Quell-Karte gelöscht wird.

Um speziell das Elementary File EF_INTERNAL und die Schlüssel K_{LVASP} und K_{RVASP} lesen zu können, überschreibt das Serviceterminal nach einer externen Authentifizierung mit dem Schlüssel K_{SO} (analog wie bei der Operation Löschen einer VAS-Applikation) zuerst die Schlüssel K_{LVASP} und kann dann nach einer erneuten Authentifizierung mit K_{LVASP} die Daten aus EF_INTERNAL übertragen.

Zusätzlich zu den VAS-Applikationen der Implementierungsklassen DF_PT und DF_AD muß die Datei EF_TRANSFER übertragen werden. Dazu werden mit der Operation Entnehmen nacheinander die Objekte dieser Implementierungsklasse entfernt, die nicht bereits als entfernt oder verfallen markiert sind, ihre Signatur mit K_{SIG_VASC} geprüft und in das EF_TRANSFER der Ziel-Karte übertragen. Auf der Ziel-Karte werden allerdings die Objekte als nicht entnommen gekennzeichnet, damit sie weiterhin gültig bleiben.

Schließlich müssen noch die globalen Daten des VAS-Containers übertragen werden. Insbesondere muß der Sequenzzähler der Ziel-Karte auf den Wert der Quell-Karte eingestellt werden.

Nachfolgend das Verfahren des Aufbringens von VAS-Daten:

Es gibt drei mögliche Fälle für das Aufbringen von VAS-Daten an einem Händlerterminal, die von der Art der VAS-Applikation abhängen.

Zunächst das Erwerben im Falle der Implementierungsklasse DF_PT oder DF_AD

Es sollen von einem VAS-Provider Daten in eine VAS-Applikation A der Implementierungsklasse DF PT oder DF_AD geschrieben werden.

Ablauf des Aufbringens von VAS-Daten:

1. Der Karteninhaber führt eine VAS-Karte in ein Händlerterminal ein.
2. Das Händlerterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID).
3. Es wird die Echtheit des VAS-Containers geprüft.

Ist das Händlerterminal selbst in Besitz des Masterkeys KGK_{AUT}, kann es eine interne Authentifizierung des VAS-Containers verlangen:

- INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{AUT}>

Ein Händlerterminal (welches nicht über den KGK_{AUT} verfügt) kann indirekt die Echtheit des VAS-Containers durch die Echtheitsprüfung der VAS-Applikation A feststellen. Denn eine VAS-Applikation kann nur an einem Serviceterminal geladen worden sein, die beim Ladevorgang die Echtheit des VAS-Containers geprüft hat. Die Echtheitsprüfung der VAS-Applikation A kann auf den Test des Händlerterminals zurückgeführt werden, ob der VAS-Container den Schlüssel K_{LVASP} oder den K_{RVASP} für die VAS-Applikation A enthält. Dies kann das Händlerterminal z. B. kontrollieren durch:

- SELECT FILE <PIX_A> (Fehlermeldung, falls VAS-Applikation A nicht in den VAS-Container geladen ist)
- INTERNAL AUTHENTICATE <Zufallszahl, KID von K_{RVASP} oder K_{LVASP}>.

Das Händlerterminal prüft die Antwort und bricht im Fehlerfall (mit Fehlermeldung) ab.

4. Das Händlerterminal selektiert die VAS-Applikation A, authentisiert sich und beschreibt die Elementary Files, die für die Abwicklung der Transaktion notwendig sind:
   - SELECT FILE <PIX_A> (entfällt, wenn bereits in Schritt 3 durchgeführt)
   - GET CHALLENGE
   - EXTERNAL AUTHENTICATE <K_{LVASP} (Zufallszahl), KID von K_{LVASP}>
   - optional: UPDATE RECORD <SFI von EF_INFO des DF_X, Daten>
   - optional: UPDATE RECORD <SFI von EF_INTERNAL des DF_X, Daten>
   - optional: UPDATE RECORD <SFI von EF_VALUE des DF_X, Daten>.

Als nächstes das Erwerben im Falle der Implementierungsklasse EF_TRANSFER

Speziell für VAS-Applikationen der Implementierungsklasse EF_TRANSFER (Applikationsklasse Gutschein) muß ein VAS-Provider keine proprietäre Dateistruktur DF_X für seine Applikation belegen. Die Folge ist, daß ein Karteninhaber nicht vor der Nutzung einer VAS-Applikation Gutschein an ein Serviceterminal gehen muß, um eine VAS-Applikation zu laden. Er kann dagegen direkt am Händlerterminal sich einen Gutschein oder eine Quittung ausstellen und diese an einem anderen Terminal erstatten (durch die Operation Entnehmen) lassen oder den Gutschein bzw. die Quittung nur einfach vorweisen (durch Lesen). Durch das Aufbringen von VAS-Daten erfolgt somit ein implizites Laden von VAS-Applikationen der Implementierungsklasse EF_TRANSFER. Für diese Applikationsklasse existiert die Implementierungsklasse EF_TRANSFER, die aus einzelnen Objekten der Art Record besteht. Ein Eintrag in EF_TRANSFER ist ausschließlich durch das Kommando TRANSFER möglich. Das Händlerterminal muß dazu im Besitz eines gültigen Entwerter-Schlüssels K_{DEC} sein und über eine PIX_A der VAS-Applikation A verfügen.

Ablauf des Aufbringens von VAS-Daten:

1. Der Karteninhaber führt eine VAS-Karte in ein Händlerterminal ein.
2. Das Händlerterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container Nummer).
3. Das Händlerterminal liest die Sequenznummer, die zusätzlich in den MAC aus Schritt 4 eingeht:
   - READ RECORD <SFI von EF_SEQ, 0> (Auslesen der Sequenznummer).
4. Mit dem Kommando TRANSFER wird in EF_TRANSFER ein Record geschrieben:
   - TRANSFER <Transaktionsdatum, Verfallsdatum, Erzeugercode, Daten, PIX_A, MAC mit K_{DEC}>
5. Das Händlerterminal kann durch Prüfung der Antwortnachricht des TRANSFER Kommandos prüfen, ob der VAS-Container echt ist (d. h. in Besitz des gemeinsamen Geheimnisses K_{DEC} ist). Wir verweisen hierzu auch auf die vorher beschriebene Antwortnachricht des Kommandos TRANSFER.

Schließlich das Erwerben von VAS-Daten durch Entwerten

Ein VAS-Provider kann durch einen Entwertevorgang mit dem Kommando TRANSFER im Transferfeld EF_TRANSFER ein Anrecht (z. B. Gutschein oder Quittung) erzeugen, das von einem anderen VAS-Provider weiter verwertet werden kann. Entwertet werden Daten aus dem EF_VALUE bzw. EF_INFO der VAS-Applikation A des entwertenden VAS-

Provider. Der Karteninhaber erwirbt in Form eines Objektes in EF_TRANSFER das Anrecht.

Ablauf des Aufbringens von VAS-Daten:

1. Der Karteninhaber führt eine VAS-Karte in ein Händlerterminal ein.
2. Das Händlerterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID).
3. Das Händlerterminal liest die Sequenznummer, die zusätzlich in den MAC aus Schritt 4 eingeht:
   - READ RECORD <SFI von EF_SEQ, 0> (Auslesen der Sequenznummer).
4. Das Händlerterminal selektiert die VAS-Applikation A:
   - SELECT FILE <PIX_A> (Fehlermeldung, falls VAS-Applikation A nicht in den VAS-Container geladen ist).
5. Das Händlerterminal nutzt das Kommando TRANSFER zum Entwerten der Daten aus dem EF_VALUE bzw. dem EF_INFO.
   - TRANSFER <Daten, MAC mit K_{DEC}>.

   Die Zusammensetzung der Kommandonachricht von TRANSFER wurde bereits beschrieben. Die Berechtigung zum Entwertevorgang erlangt das Terminal, wenn es eine korrekte Signatur über die Daten mit dem Schlüssel K_{DEC} bilden kann. Diese wird durch den VAS-Container geprüft. Bei Erfolg wird vom VAS-Container ein Record in EF_TRANSFER angelegt sowie die Sequenznummer inkrementiert.
6. Das Händlerterminal kann durch Prüfung der Antwortnachricht des TRANSFER Kommandos prüfen, ob der VAS-Container echt ist (d. h. in Besitz des gemeinsamen Geheimnisses K_{DEC} ist).

Und nun noch das Verfahren zum Entwerten von VAS-Daten. Es gibt zwei Arten von Entwertungsvorgängen:

Zum einen können VAS-Daten für VAS-Applikationen der Implementierungsklasse DF_PT oder DF_AD durch den zugehörigen VAS-Provider entwertet werden durch die Operation Entwerten, d. h. Erwerben von VAS-Daten durch Entwerten. Dadurch wird ein Wert verbraucht, wobei ein potentielles Anrecht auf einen weiteren Wert entsteht.

Zum anderen können VAS-Daten aus dem EF_TRANSFER einmalig entnommen werden mit dem Ergänzungskommando TAKE. In diesem Fall wird das Anrecht aufgebraucht, die Daten bleiben für etwaige weitere Verwendung (z. B. rückerstattetes Ticket wird zur Rückfahrt noch benötigt) im Transferfeld als entnommen gekennzeichnet stehen, bis sie bei Bedarf von anderen Objekten überschrieben werden.

Ablauf des Entwertens von VAS-Daten durch das Kommando TAKE:

1. Der Karteninhaber führt eine VAS-Karte in ein Händlerterminal ein.
2. Das Händlerterminal prüft, ob ein VAS-Container vorhanden ist:
   - SELECT FILE <AID_{VAS}> (Fehlermeldung wenn VAS-Container nicht selektierbar)
   - READ RECORD <SFI von EF_ID, 0> (Auslesen der VAS-Container ID).
3. Zunächst liest das Händlerterminal die zur Verfügung stehenden Objekte aus EF_TRANSFER aus mit dem Spezialfall der Operation Ansehen VAS-Applikationen, um zu entscheiden, ob ein gesuchtes Objekt enthalten ist. Alternativ kann mit dem Kommando SEEK nach einem Muster gesucht werden. Das Terminal kennt im Erfolgsfall die Nummer i des gesuchten Records.
4. Das Terminal markiert den Record mit Recordnummer i als entfernt, indem es i, eine von ihm berechnete Zufallszahl, seine PIX und Terminal ID mit dem Kommando TAKE übermittelt:
   - TAKE <i, Zufallszahl, PIX, Terminal ID>.

Das Händlerterminal nutzt das Kommando TAKE zum Lesen der Daten aus dem EF_TRANSFER bei gleichzeitigem Kennzeichnen der Daten als entnommen. Die Ausführung des Kommandos bewirkt zusätzlich die Erzeugung von zwei verschiedenen Kryptogrammen C_1 und C_2.

Das Kryptogramm C_1 wird durch den VAS-Container mit dem Schlüssel K_{SIG_VASC} berechnet. Damit kann der Einreicher des entnommenen, mit C_1 signierten Objektes sich beim System Operator die Einmaligkeit und Echtheit nachweisen lassen. Die Einmaligkeit und Echtheit der Transaktion ergibt sich aus dem Kryptogramm des VAS-Providers, von dem das Objekt ursprünglich stammt (und die von der Karte geprüft wurde, siehe TRANSFER), und der Führung eines Sequenzzählers über die Transaktionen sowie dem Kryptogramm C_1 durch die Karte bei der Entnahme.

Das Kryptogramm C_2 wird durch den VAS-Container mit dem Schlüssel K_{DEC} berechnet, der von dem VAS-Container aus KGK_{DEC}, PIX und Terminal ID abgeleitet wird. Durch Kenntnis des gemeinsamen Geheimnisses K_{DEC} kann der VAS-Container dem Terminal gegenüber direkt die Echtheit des VAS-Containers beweisen. Da beim TRANSFER Kommando ebenfalls geprüft wird, daß nur echte Gutscheine bzw. Quittungen in einen echten VAS-Container gespeichert werden, kann daraus die Echtheit des entnommenen Objektes gefolgert werden. Da C_2 indirekt über die Sequenznummer, das Entnommen-Bit und die VAS-Container ID gebildet wird, kann der VAS-Container dem Terminal sogar die Einmaligkeit des entnommenen Objektes nachweisen.

Die Berechtigung zum Entnehmen mit dem Kommando TAKE hat jeder.

Am Serviceterminal ist ferner die Deaktivierung bzw. Aktivierung eines Paßwort- oder PIN-Schutzes für das Lesen von VAS-Applikationen der Implementierungsklassen DF_PT und DF_AD möglich. Außerdem kann die PIN des VAS-Containers vom Karteninhaber durch der Kenntnis der PIN geändert werden und vom System Operator mittels externer Authentifizierung durch K_{SO} zurückgesetzt werden. Als PIN oder Paßwort sind auch nicht-numerische Zeichen oder Zeichenketten beliebiger Länge verwendbar.

Patentansprüche

1. Chipkarte, die zur Durchführung von Transaktionen dient, bei denen geldwerte Einheiten oder andere nicht-mo-

netäre Ansprüche repräsentierende Wertdaten zwischen dem Karteninhaber und mindestens einem Transaktionspartner (Service-Provider) übertragen oder dem Service-Provider zur Verifizierung der Ansprüche vorgewiesen werden, wobei die Chipkarte einen Speicher umfaßt, in dem zur Durchführung der Transaktionen erforderliche Daten abgespeichert werden, und die Chipkarte **dadurch gekennzeichnet** ist, daß sie folgendes aufweist:

eine Einrichtung zum Laden von einer oder mehreren Kartenanwendungen (VAS-Applikationen) auf die Karte, die jeweils die Durchführung von Transaktionen zwischen dem Karteninhaber und einem oder mehreren Service-Provider ermöglichen.

2. Chipkarte nach Anspruch 1, bei der die Einrichtung zum Laden folgendes aufweist:

eine darin abgespeicherte Datenstruktur (DF_VAS, VAS-Container), die aufweist:

eine Teilstruktur (DF_PT, DF_AD, VAS-Applikation), in die zur Ermöglichung der Durchführung einer Transaktion zwischen dem Karteninhaber und einem Service-Provider erforderliche Daten (VAS-Daten) geladen werden können;

einen Definitionsdatensatz, der Informationen über die Art und/oder Struktur der in der Teilstruktur (VAS-Applikation) abgespeicherten Daten enthält; wobei der Definitionsdatensatz ferner aufweist:

eine die Datenstruktur (VAS-Container) und/oder die Chipkarte identifizierende Kennung (Container-ID); und

mindestens einen System-Schlüssel (K_{SO}), der die Integrität des Definitionsdatensatzes und/oder der Datenstruktur (VAS-Container) gegen Modifikationen absichert.

3. Chipkarte nach Anspruch 1 oder 2, dadurch gekennzeichnet, daß sie ferner aufweist:

einen Transfer-Speicherbereich (EF_TRANSFER) zur Aufnahme von bei der Durchführung der Transaktion auszutauschenden oder vorzuweisenden die geldwerte Einheiten oder nicht monetären Ansprüche repräsentierenden Daten; und

eine Einrichtung zum Schreiben von Daten in den Transferspeicher in Reaktion auf ein Schreibkommando (Transfer).

4. Chipkarte nach einem der Ansprüche 1 bis 3, dadurch gekennzeichnet, daß sie weiterhin mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zum Laden von für die Durchführung von Transaktionen erforderlichen Daten in die Teilstruktur unter Verwendung des mindestens einen Systemschlüssels;

eine Einrichtung zum Schreiben von Daten in den Definitionsdatensatz zum Anpassen der Daten des Definitionsdatensatzes an die in die Teilstruktur geladenen Daten;

eine Einrichtung zum Generieren einer weiteren Teilstruktur, in die zur Durchführung einer Transaktion erforderliche Daten geladen werden können, in dem Speicher der Karte; und/oder

eine Einrichtung zur dynamischen Speicherverwaltung auf der Karte.

5. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß

es sich bei den Teilstrukturen (VAS-Applikationen) um voneinander unabhängige Teilstrukturen handelt, die jeweils einem bestimmten Service-Provider zugeordnet sind,

der Definitionsdatensatz mittels des mindestens einem Systemschlüssels (K_{SO}) gegen Modifikationen geschützt ist und nur mittels dieses Schlüssels modifiziert werden kann, und daß

das Laden der Teilstrukturen (VAS-Applikationen) nur unter Verwendung des im Definitionsdatensatz enthaltenen Systemschlüssels erfolgen kann.

6. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß der Definitionsdatensatz ferner mindestens eines der folgenden Merkmale aufweist:

mindestens einen Authentifizierungsschlüssel (K_{AUT}) zur Authentifizierung der Berechtigung der Chipkarte gegenüber einem Terminal und/oder zur Authentifizierung der Berechtigung eines Terminals gegenüber der Chipkarte;

mindestens einen Signierschlüssel (K_{SIG_VASC}) zum Signieren von aus dem Transferspeicher entnommenen Daten;

einen Schlüsselerzeugungs-Schlüssel (KGK_{DEC}) zur Erzeugung von terminal- und applikationsspezifischen Schlüsseln zur Überprüfung der Berechtigung eines Schreibvorgangs in den Transferspeicher und/oder einer Entwertung von Wertdaten;

eine PIN zur Verifikation der Berechtigung eines Transaktionsvorgangs durch den Karteninhaber,

daß der Systemschlüssel (K_{SO}), der Authentizifierungsschlüssel (K_{AUT}), der Signierschlüssel (K_{SIG_VASC}) und der Schlüsselerzeugungs-Schlüssel (KGK_{DEC}) kartenindividuelle oder datenstruktur-(DF_VAS)-spezifische Schlüssel sind, und daß die Teilstruktur (VAS-Applikation) mindestens eines der folgenden Merkmale aufweist:

mindestens einen Wertspeicher (EF_VALUE) zur Aufnahme von Wertdaten;

mindestens einen Intern-Speicher (EF_INTERNAL) zur Aufnahme von internen die Teilstruktur betreffenden Daten;

mindestens einen Infospeicher (EF_INFO) zur Aufnahme von nicht internen die Teilstruktur betreffenden Daten;

einen Schlüsselspeicher (EF_KEY) zur Aufnahme mindestens eines Schlüssels (K_{LVASP}, K_{RVASP}), die Schreib- und/oder Lesevorgänge in den oder aus dem Wertspeicher, und/oder dem Intern-Speicher und/oder dem Info-Speicher absichern, und daß die Chipkarte ferner umfaßt:

eine Einrichtung zum Schreiben und/oder Lesen von Daten in den oder aus dem Wertspeicher, dem Intern-Speicher und dem Infospeicher, und daß die Einrichtung zum Laden umfaßt:

eine Einrichtung zum Schreiben der Schlüssel in den Schlüsselspeicher unter Absicherung durch den mindestens einen Systemschlüssel.

7. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Teilstruktur mindestens eines der folgenden Merkmale aufweist:

einen Schlüssel (K_{LVASP}) zum Überprüfen der Schreibberechtigung von Daten in die Teilstruktur;

einen Schlüssel (K_{RVASP}) zum Überprüfen der Leseberechtigung von Daten aus der Teilstruktur.

8. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die im Schlüsselspeicher abgespeicherten Schlüssel für die jeweilige Teilstruktur spezifisch sind und daß die mindestens eine Teilstruktur

(VAS-Applikation) das Durchführen von Transaktionen mittels mindestens eines der spezifischen Schlüssel (K_{LVASP}, K_{RVASP}) absichert, der für die jeweilige Teilstruktur (VAS-Applikation) spezifisch und von den Schlüsseln anderer Teilstrukturen (VAS-Applikationen) unabhängig ist.

9. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß
die Karte mehrere der Teilstrukturen (VAS-Applikationen) aufweist, die jeweils zur Durchführung von Transaktionen zwischen einem bestimmten Service-Provider und dem Karteninhaber dienen, und daß
das Durchführen von Transaktionen das Schreiben und/oder Lesen von Daten in das bzw. aus dem Transferfeld und/oder das Schreiben und/oder Lesen von Daten in den bzw. aus dem Wertspeicher umfaßt.

10. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß sie ferner umfaßt:
eine Einrichtung zum Durchführen von Transaktionen sowohl zwischen einzelnen Teilstrukturen (VAS-Applikationen) als auch zwischen einer Teilstruktur und einem Service-Provider.

11. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß
der System-Schlüssel (K_{SO}) nur dem Systembetreiber (System Operator SO) bekannt ist und ein kartenindividueller und/oder Datenstruktur-(VAS-Container)-spezifischer Schlüssel ist, und daß
die weiteren im Definitionsdatensatz enthaltenen Schlüssel kartenindividuelle und/oder Datenstruktur-(VAS-Container)-spezifische Schlüssel sind.

12. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß der Definitionsdatensatz eines oder mehrere der folgenden Merkmale aufweist:
eine die Datenstruktur spezifizierende Identifikationsnummer (EF_ID) ein Verzeichnis der in der Datenstruktur enthaltenen Teilstrukturen (EF_DIR), wobei das Verzeichnis teilstrukturspezifische Identifikationsnummern von in der Datenstruktur (VAS-Container) geladenen Teilstrukturen (VAS-Applikationen) enthält, sowie Informationen über den Teil der Datenstruktur (VAS-Container), in dem die Teilstrukturen (VAS-Applikationen) physikalisch gespeichert sind;
eine Versionsnummer der Datenstruktur (EF_VERSION).

13. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß sie mindestens eines der folgenden Merkmale umfaßt:
eine Einrichtung zur Durchführung eines Entnahmevorgangs (Take), mittels dessen Daten aus dem Transferspeicher entnommen und/oder entwertet werden;
eine Einrichtung zur Erzeugung eines oder mehrerer Echtheitsmerkmale der Daten bei Entnahme bzw. der Entwertung von Daten aus dem Transferspeicher.

14. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Chipkarte zur Generierung der Echtheitsmerkmale folgendes aufweist:
einen Signierschlüssel K_{SIG_VASC} zum Erzeugen einer digitalen Signatur über den entnommenen Daten;
eine Einrichtung zur Erzeugung einer die Transaktion kennzeichnenden Transaktionsnummer, die zur Erzeugung der digitalen Signatur mitverwendet wird.

15. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß der Signierschlüssel K_{SIG_VASC} ein aus einem privaten Key-Generating-Key abgeleiteter privater Schlüssel ist und zur Überprüfung der Signatur durch die Serviceprovider öffentliche Schlüssel herangezogen werden.

16. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß sie mindestens eines der folgenden Merkmale umfaßt:
eine Einrichtung zur Erzeugung von terminal- und teilstrukturspezifischen Schlüsseln (K_{DEC}) mittels des Schlüsselerzeugungs-Schlüssels (KGK_{DEC});
eine Einrichtung zur Verifizierung der Rechtmäßigkeit und/oder der Absicherung einer Transaktion unter Verwendung mindestens eines der folgenden Merkmale:
des terminal- und teilstrukturspezifischen Schlüssels (K_{DEC}),
des mindestens einen Authentifizierungsschlüssels (K_{AUT}),
des mindestens einen Systemschlüssels (K_{SO}),
des mindestens einen teilstrukturspezifischen Schlüssels (K_{LVASP}, K_{RVASP}),
des Signierschlüssels (K_{SIG_VASC}),
der PIN,
der Kennung einer Teilstruktur,
der Kennung eines Terminals.

17. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Chipkarte ferner mindestens eines der folgenden Merkmale umfaßt:
eine Einrichtung zum Authentifizieren der Berechtigung und/oder des Terminals mittels des Authentifizierungsschlüssels bevor ein Lese- oder Schreibvorgang gestartet wird,
eine Einrichtung zur Durchführung von Lese- oder Schreibvorgängen, die mit einer digitalen Unterschrift und/oder Verschlüsselung gesichert sind.

18. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Chipkarte ferner mindestens eines der folgenden Merkmale umfaßt:
eine Einrichtung zum Aktivieren und Deaktivieren des PIN-Schutzes,
eine Einrichtung zum Abändern der PIN.

19. Chipkarte nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß
die Datenstruktur gemäß einem der vorhergehenden Ansprüche von der Kartenplattform unabhängig ist und die Chipkarte ferner umfaßt:
eine Einrichtung zum Übertragen der Datenstruktur oder von Teilen der Datenstruktur auf eine andere Karte.

20. Chipkarte, dadurch gekennzeichnet, daß sie folgendes aufweist:
einen Speicherbereich zum Schreiben oder Lesen von Daten in den oder aus dem Speicherbereich zum Transfer von

geldwerten Einheiten und/oder von anderen nicht-monetären Ansprüche repräsentierenden Wertdaten zwischen identischen oder unterschiedlichen Service-Providern.

21. Terminal zur Verwendung mit einer Chipkarte gemäß einem der Ansprüche 1 bis 20, dadurch gekennzeichnet, daß das Terminal umfaßt:

eine Einrichtung zum Identifizieren der Datenstruktur (VAS-Container) der Chipkarte sowie zur Identifikation der die Datenstruktur identifizierenden Kennung (Container-ID);

und daß es ferner mindestens eines der folgenden Merkmale umfaßt:

eine Einrichtung zum Lesen von Daten aus mindestens einer der Teilstrukturen und/oder dem Definitionsdatensatz und/oder dem Transferspeicher der Karte;

eine Einrichtung zum Schreiben von Daten in den Transferspeicher der Karte;

eine Einrichtung zum Laden der zur Durchführung von Transaktionen erforderlichen Daten in mindestens eine der Teilstrukturen (VAS-Applikationen) der Karte.

22. Terminal nach Anspruch 21, dadurch gekennzeichnet, daß es ferner mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zur Durchführung von Transaktionen zwischen einem Service-Provider und dem Karteninhaber, wobei das Durchführen einer Transaktion mindestens einen der folgenden Schritte umfaßt:

das Schreiben von Daten in den Wertspeicher,

das Schreiben von Daten in den Transferspeicher,

das Entnehmen und/oder Entwerten von Daten aus dem Transferspeicher,

das Lesen von Daten aus der Teilstruktur,

das Lesen von Daten aus dem Transferspeicher.

23. Terminal nach einem der Ansprüche 21 oder 22, dadurch gekennzeichnet, daß es ferner umfaßt:

eine Einrichtung zur Verifizierung der Rechtmäßigkeit und/oder der Absicherung einer Transaktion unter Verwendung mindestens eines der folgenden Merkmale:

eines terminal- und teilstrukturspezifischen Schlüssels (K_{DEC}),

des mindestens einen kartenindividuellen oder datenstruktur-(DF_VAS)-spezifischen Authentifizierungsschlüssels (K_{AUT}),

des mindestens einen kartenindividuellen oder datenstruktur-(DF_VAS)-spezifischen Systemschlüssels (K_{SO}),

des mindestens einen teilstrukturspezifischen Schlüssels (K_{LVASP}, K_{RVASP}),

des kartenindividuellen oder datenstruktur-(DF_VAS)-spezifischen Signierschlüssels (K_{SIG_VASC}),

der kartenindividuellen oder datenstruktur-(DF_VAS)-spezifischen PIN,

der applikationsspezifischen Kennung einer Teilstruktur,

der terminalspezifischen Kennung eines Terminals.

24. Terminal nach einem der Ansprüche 219 bis 23, dadurch gekennzeichnet, daß die Einrichtung zum Schreiben von Daten in den Transferspeicher aufweist:

eine Einrichtung zum Verschlüsseln von Daten unter Verwendung eines terminal- und teilstrukturspezifischen Schlüssels (K_{DEC}) zum Nachweis der Schreibberechtigung.

25. Terminal nach einem der Ansprüche 21 bis 24, dadurch gekennzeichnet, daß es ferner aufweist:

eine Einrichtung zum Durchführen eines Entnahmevorgangs von Daten aus dem Transferspeicher, mittels dessen Daten aus dem Transferspeicher entnommen und/oder entwertet werden.

26. Terminal nach einem der Ansprüche 21 bis 25, dadurch gekennzeichnet, daß es ferner mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zum Kennzeichnen von Daten des Transferspeichers als entnommen,

eine Einrichtung zum Kennzeichnen von Daten des Transferspeichers als verfallen.

27. Terminal nach einem der Ansprüche 21 bis 26, dadurch gekennzeichnet, daß die Einrichtung zum Durchführen von Transaktionen mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zum Ändern von Wertdaten in der Teilstruktur;

eine Einrichtung zum Durchführen von Transaktionen zwischen verschiedenen Service-Providern (Interservices) auf Kosten bzw. zum Nutzen des Karteninhabers.

28. Terminal nach einem der Ansprüche 21 bis 27, dadurch gekennzeichnet, daß es ferner aufweist:

eine Einrichtung zum Authentifizieren der Berechtigung des Terminals gegenüber der Karte und/oder der Karte gegenüber dem Terminal unter Verwendung mindestens eines Authentifizierungsschlüssels;

eine Einrichtung zum Absichern einer Transaktion durch den Karteninhaber mittels einer PIN;

eine Einrichtung zum Aktivieren und Deaktivieren des PIN-Schutzes.

29. Terminal nach einem der Ansprüche 21 bis 28, dadurch gekennzeichnet, daß es ferner mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zum Übertragen einer terminalspezifischen Kennung an die Karte;

eine Einrichtung zum Übertragen einer die Teilstruktur spezifizierenden Kennung an die Karte;

eine Einrichtung zum Authentifizieren der Berechtigung unter Verwendung eines terminal- und teilstrukturspezifischen Schlüssels sowie der terminal- und teilstrukturspezifischen Kennung,

eine Einrichtung zur Durchführung von Lese- oder Schreibvorgängen, die mit einer digitalen Unterschrift und/oder Verschlüsselung gesichert sind.

30. Terminal nach einem der Ansprüche 21 bis 29, dadurch gekennzeichnet, daß es ferner mindestens eines der folgenden Merkmale aufweist:

eine Einrichtung zum Selektieren einer Teilstruktur (VAS-Applikation);

eine Einrichtung zum Ansehen einer Teilstruktur auf dem Terminal;

eine Einrichtung zum Ansehen der Daten einer Teilstruktur auf dem Terminal;

eine Einrichtung zum Laden einer Teilstruktur (VAS-Applikation) auf die Karte;

eine Einrichtung zum Laden einer Kennung einer geladenen Teilstruktur (VAS-Applikation) auf die Karte,
eine Einrichtung zum Löschen einer Teilstruktur von der Karte;
eine Einrichtung zum Substituieren einer Teilstruktur durch eine andere Teilstruktur;
eine Einrichtung zum Übertragen einer Teilstruktur auf eine andere Karte;
eine Einrichtung zum Interpretieren einer Teilstruktur bezüglich ihrer Funktion und ihres zugeordneten Service-Providers sowie zum Lesen und Ansehen von in ihr gespeicherten Informationen.

31. Verfahren zum Durchführen von Transaktionen zwischen einem Karteninhaber und mindestens einem Service-Provider unter Verwendung einer Chipkarte sowie eines Terminals, wobei das Verfahren einen der folgenden Schritte umfaßt:
Vorsehen einer auf der Chipkarte abgespeicherten Datenstruktur, in die zur Ermöglichung der Durchführung der Transaktion zwischen dem Karteninhaber und einem Service-Provider erforderliche Daten (VAS-Daten) geladen werden können, sowie Schreiben oder Lesen von Daten in die/aus der Datenstruktur (VAS-Applikation);
Vorsehen eines Transfer-Speicherbereichs (EF_TRANSFER) zur Aufnahme von bei der Durchführung der Transaktion auszutauschenden oder vorzuweisenden die Geldwerteeinheiten oder nicht-monetären Ansprüche repräsentierenden Daten, sowie Schreiben von Daten in den Transferspeicher oder Lesen von Daten aus dem Transferspeicher.

32. Verfahren nach Anspruch 31, dadurch gekennzeichnet, daß das Verfahren mindestens einen der folgenden Schritte umfaßt:
Verwendung einer Chipkarte gemäß einem der Ansprüche 1 bis 20;
Verwendung eines Terminals gemäß einem der Ansprüche 21 bis 30;
Schreiben oder Lesen von Daten in den/aus dem Wertspeicher oder Intern-Speicher oder Info-Speicher von mindestens einer der Teilstrukturen (VAS-Applikationen).

33. Verfahren nach Anspruch 31 oder 32, dadurch gekennzeichnet, daß es ferner mindestens einen der folgenden Schritte umfaßt:
Authentifizieren der Berechtigung des Terminals und/oder der Chipkarte unter Verwendung mindestens eines Schlüssels;
Absichern der Transaktion durch Verwendung einer digitalen Unterschrift und/oder Verschlüsselung durch Verwendung mindestens eines Schlüssels.

34. Verfahren zum Laden von Daten auf eine Chipkarte unter Verwendung eines Terminals, dadurch gekennzeichnet, daß das Verfahren mindestens einen der folgenden Schritte umfaßt:
Laden von Daten in eine Teilstruktur (VAS-Applikation) der Karte;
Schreiben von Daten in den Definitionsdatensatz der Karte.

35. Verfahren zum Laden von Daten nach Anspruch 34, welches mindestens einen der folgenden Schritte umfaßt:
Verwendung einer Chipkarte gemäß einem der Ansprüche 1 bis 20;
Verwendung eines Terminals gemäß einem der Ansprüche 21 bis 30.

36. System zur Durchführung von Transaktionen, gekennzeichnet durch:
eine Chipkarte gemäß einem der Ansprüche 1 bis 20 und
ein Terminal gemäß einem der Ansprüche 21 bis 30.

Hierzu 8 Seite(n) Zeichnungen

- Leerseite -

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

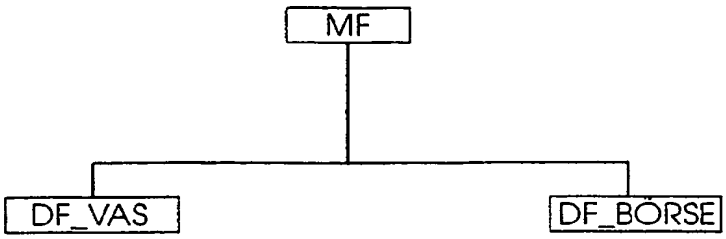


Fig. 1

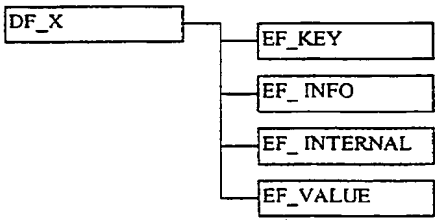


Fig. 6

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

VAS Systemüberblick

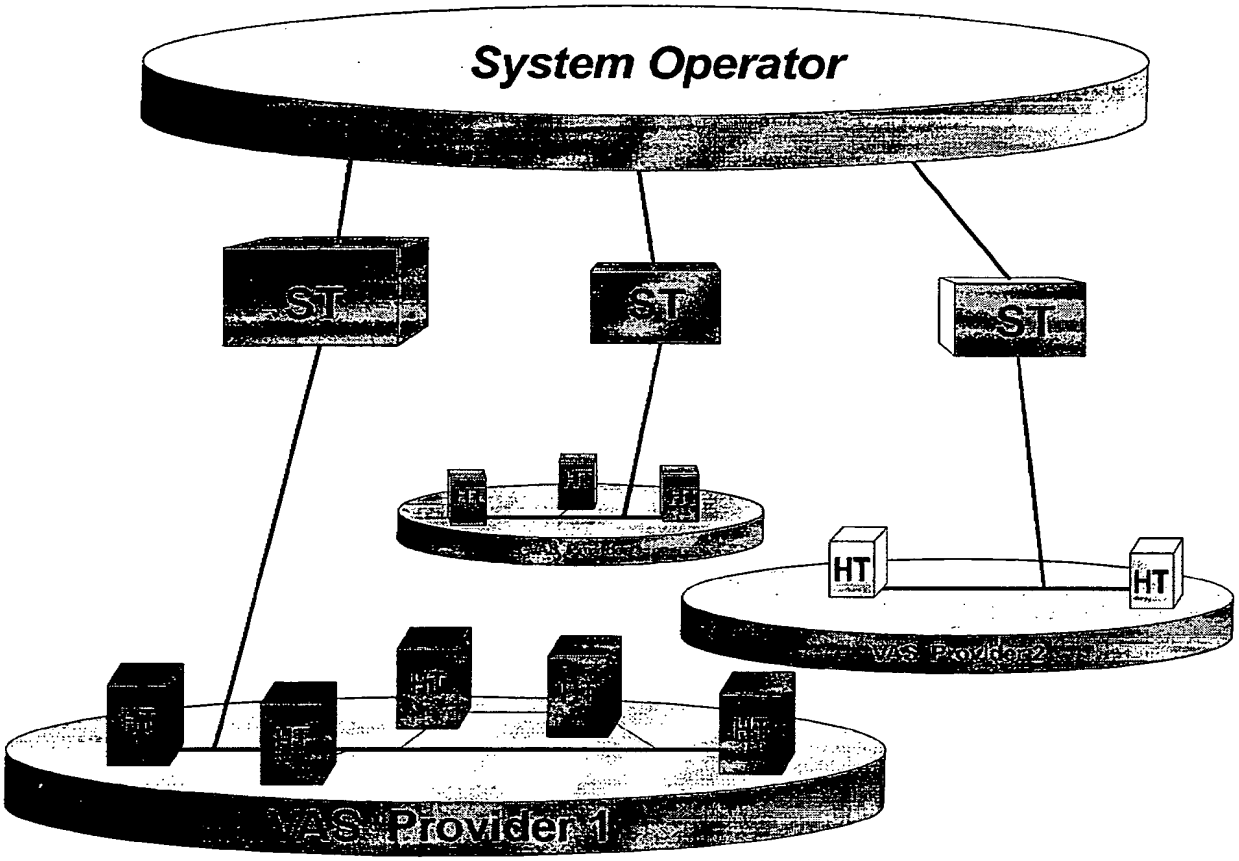


Fig. 2

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

VAS Datenfluß

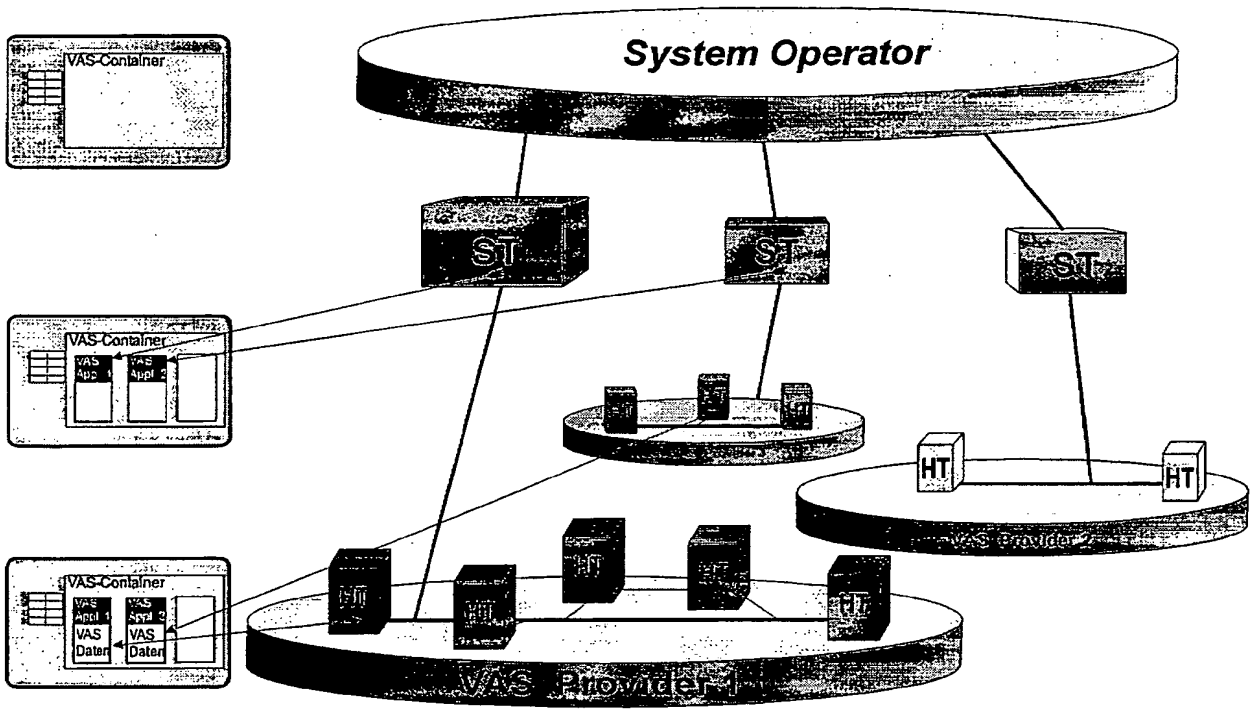


Fig. 3

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

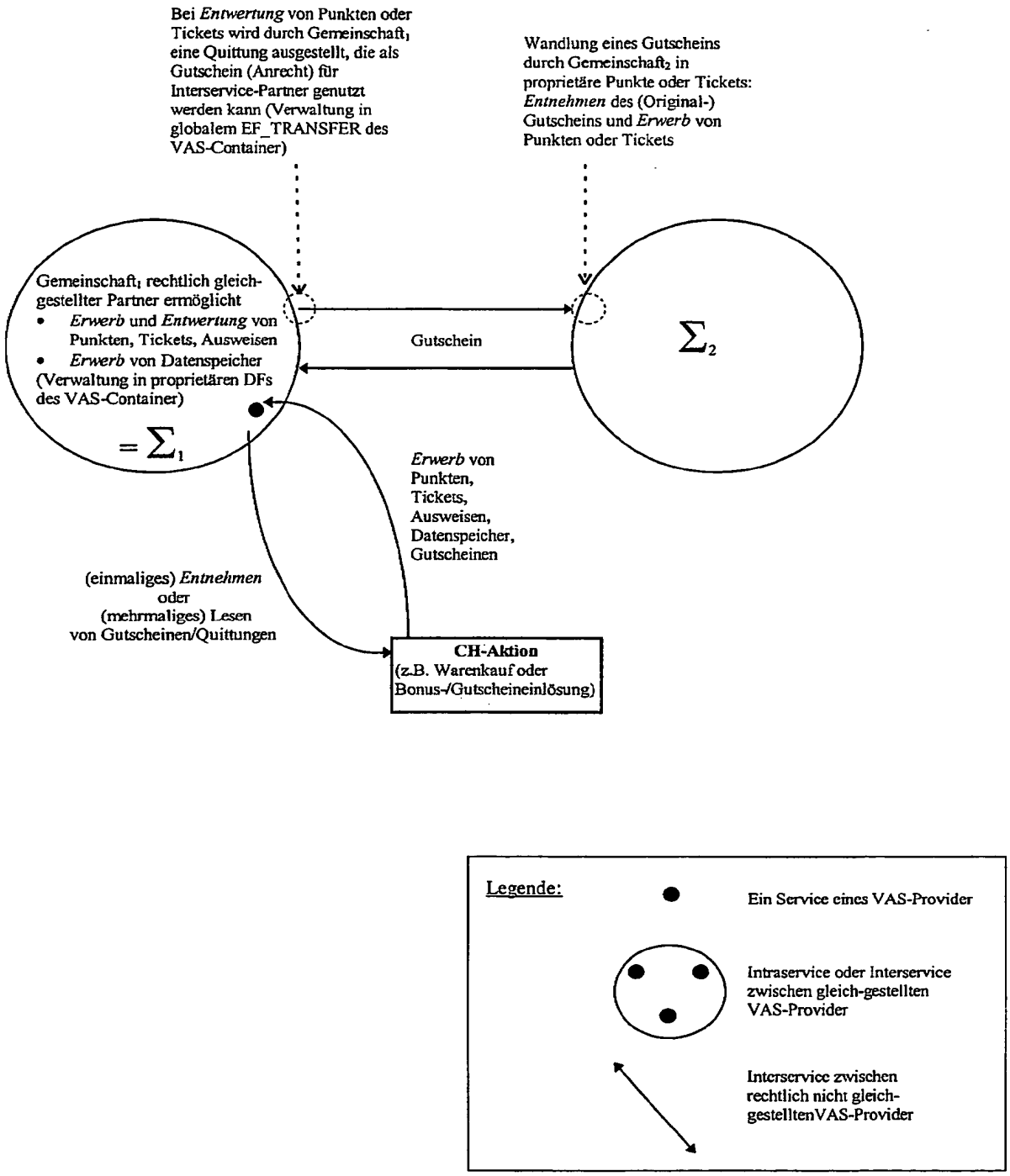


Fig. 4

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

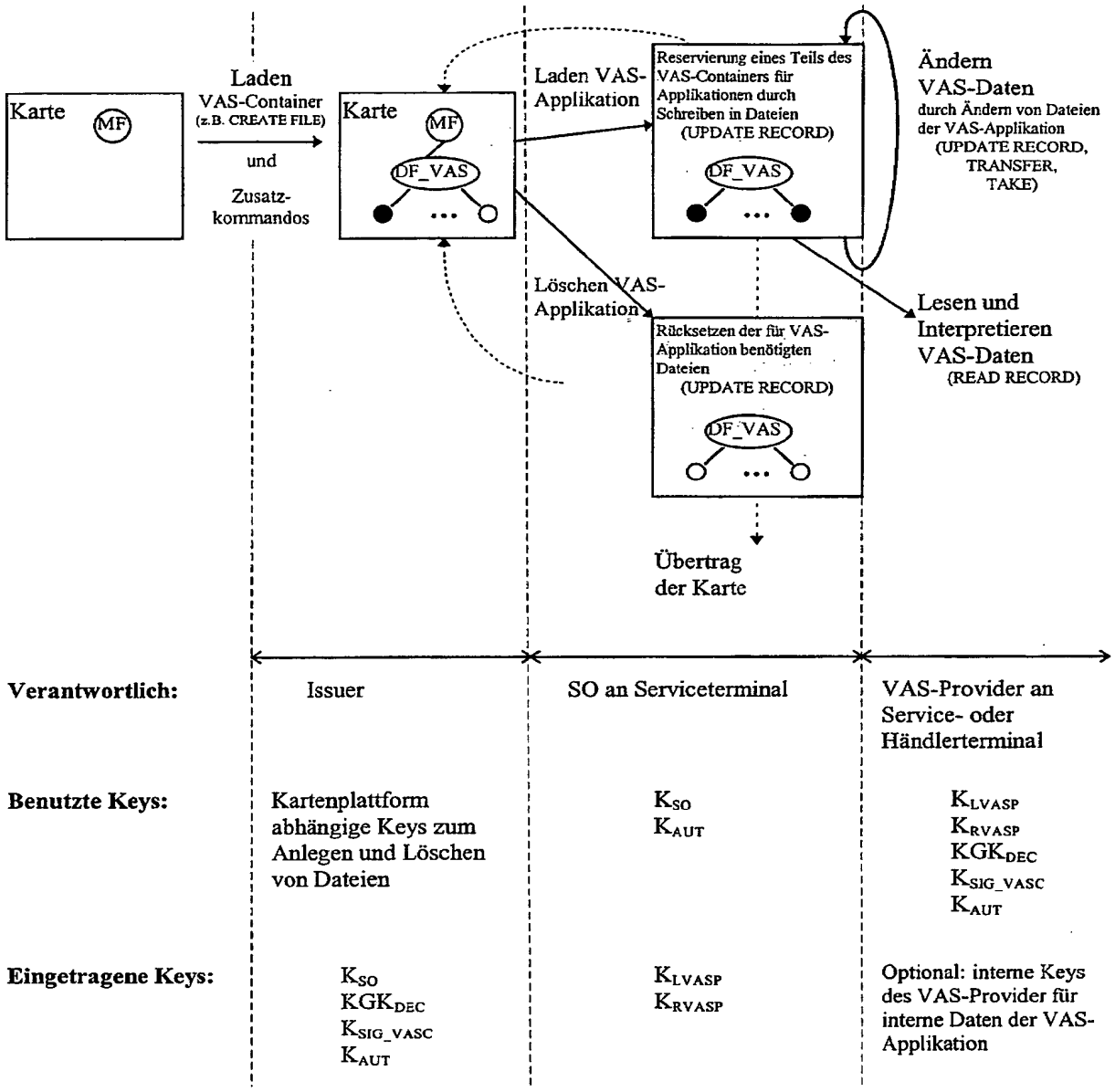


Fig. 5

802 026/900

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

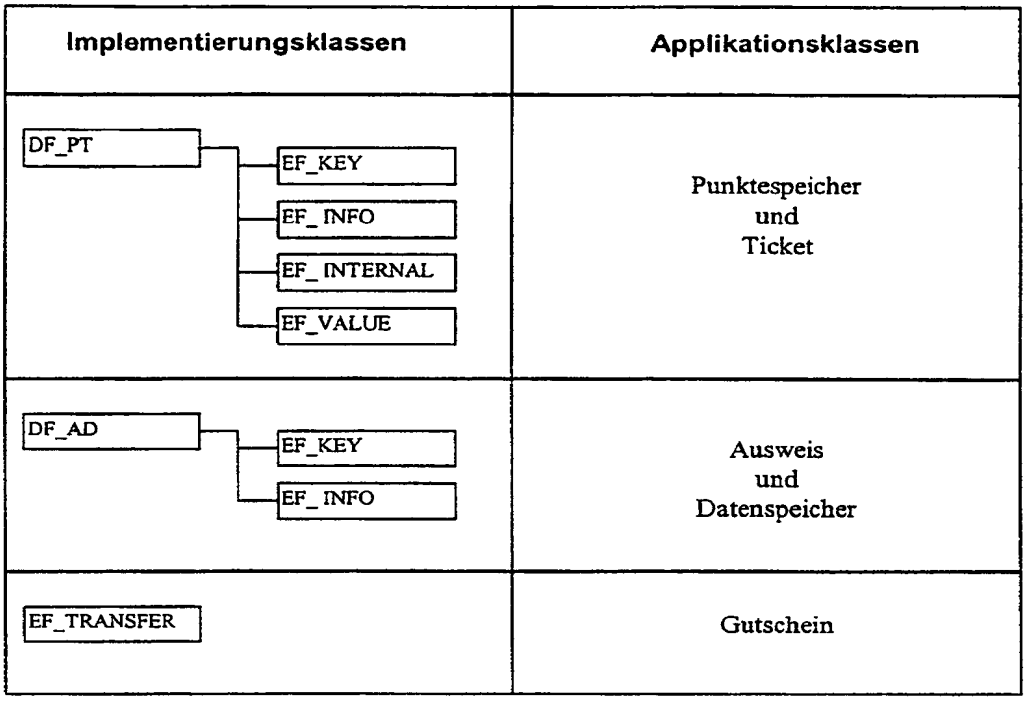


Fig. 7

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

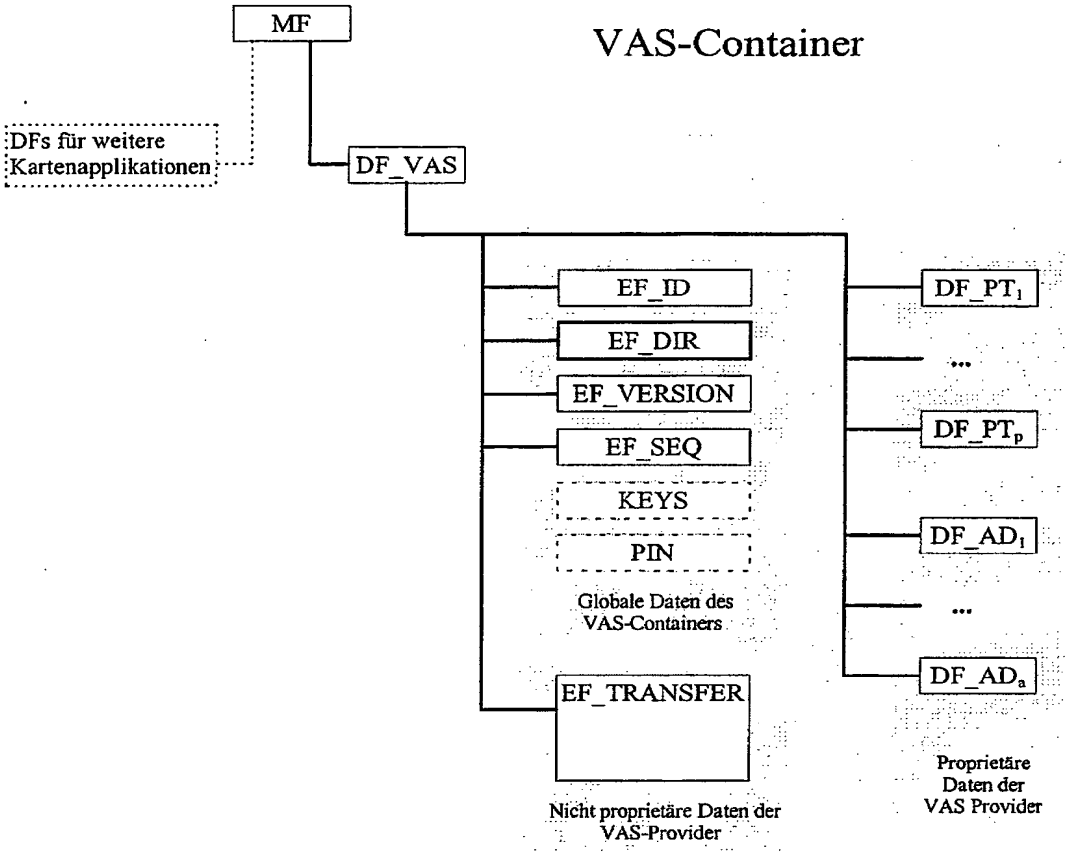


Fig. 8

Nummer: DE 197 18 115 A1
Int. Cl.[6]: G 06 K 19/07
Offenlegungstag: 25. Juni 1998

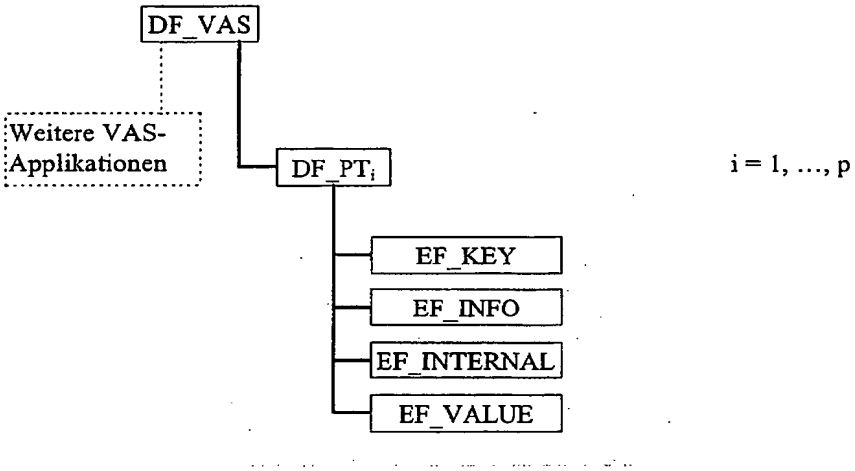


Fig. 9

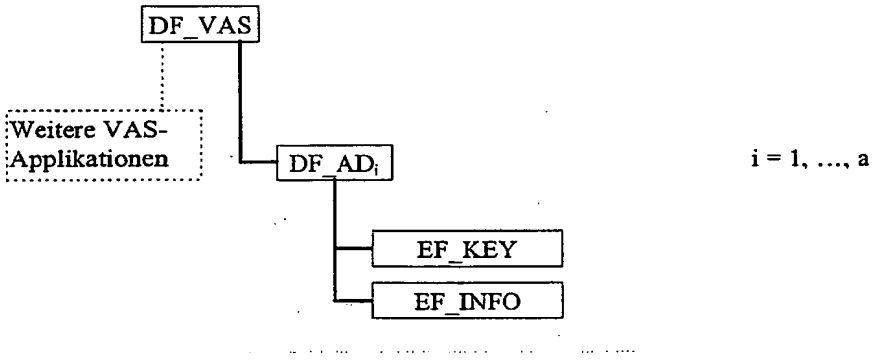


Fig. 10